Failure Detection, Isolation and Mitigation for a Spacecraft Solid Fuel Reaction Control System

by

Richard C. Shreffler

B.S. Honors Systems Engineering
United States Naval Academy, 2010

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN MECHANICAL ENGINEERING
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2012

Signature of Author: _____
Department of Mechanical Engineering
May 11, 2012

Certified by: _____
Edward Bergmann
Draper Laboratory Technical Supervisor

Certified by: _____
John Leonard
Professor of Mechanical and Ocean Engineering
Thesis Supervisor

Accepted by: _____
David Hardt
Ralph E. and Eloise F. Cross Professor of Mechanical Engineering
Chairman, Department Committee on Graduate Theses

Failure Detection, Isolation and Mitigation for a Spacecraft Solid Fuel Reaction Control System

by

Richard C. Shreffler

Submitted to the Department of Mechanical Engineering
on May 11, 2012 in Partial Fulfillment of the
Requirements for the Degree of Master of Science in
Mechanical Engineering

ABSTRACT

Future unmanned space missions will require increased redundancy to failure. One such mission is the Mars Sample Return, intended to return a sample of Martian soil, rock and atmosphere to Earth for greater study. A key element of the Mars Sample Return mission is the Mars Ascent Vehicle, which is designed to carry the sample container from the surface into Mars orbit, where the sample will be transferred to the Earth return vehicle. The Mars Ascent Vehicle is currently proposed as a two-stage solid fuel vehicle, which is ideal for surviving the long interplanetary journey and environmental extremes of the Martian surface.

This thesis details real time failure detection, isolation and mitigation algorithms for use with a six degree of freedom solid fuel reaction control system, which commonly functions by burning a solid fuel gas generator and expelling gas out of valves arranged around the vehicle. Valve failures are known to occur in spacecraft reaction control systems, and without on-board, real time failure identification and mitigation, the Mars Ascent Vehicle could fail to place the sample container in the proper orbit. The two failure modes which are considered here are valves firing continuously, an on failure, or a valve not firing when commanded, an off failure. The detection and isolation algorithm relies on comparing the expected with the actual change in vehicle angular and linear rates in order to determine the disturbance acceleration that has been applied to the vehicle. For an on failure, the mitigation algorithm works by commanding on the jet which directly opposes the failed jet, and for an off failure, the failed jet is simply removed from the available jets to be commanded.

The algorithms detailed here provide increased redundancy to failure and greater robustness without the need for additional dedicated detection hardware and at minimal computing load. These algorithms could also be adapted to other space vehicles and numerous other applications.

Thesis Supervisor: John Leonard
Title: Professor of Mechanical and Ocean Engineering

Technical Supervisor: Edward Bergmann
Title: Principal Member of the Technical Staff, Charles Stark Draper Laboratory

Acknowledgements

I wish to thank Edward Bergmann, my thesis supervisor, for identifying a problem and conceiving the idea for this research, for his invaluable advice and input throughout this project, and for his willingness to share his wealth of knowledge, particularly in regards to spacecraft flight controls. I would also like to thank Professor John Leonard, my thesis advisor, for his valuable contributions to this research and support throughout my time at MIT.

Many members of the Draper Laboratory staff provided valuable insight and participated in this research, including: Bruce Persson, Andre Schor, Sungyung Lim, Leena Singh, Onno Hommes, Paul Germain, and Ben Lane. I have been lucky enough to be a part of their ongoing work.

I also wish to thank my wife, Auden, for her tireless support.

Publication of this work does not constitute approval by the Charles Stark Draper Laboratory of the findings or conclusions contained herein. It is published only for the exchange and stimulation of ideas.

Table of Contents

1.0 Introduction

1.1 Research Motivation

As space exploration continues in the current and foreseeable budget-constrained environment, increasing emphasis will be placed on unmanned robotic exploration missions. These missions will require significant robustness to failure, both in hardware design and in control software.

One proposed mission of particular interest to the scientific community is the Mars Sample Return (MSR) mission. The three key elements of the proposed mission are: a rover or lander to collect samples from the Mars surface, a Mars Ascent Vehicle (MAV) to carry the sample into Mars orbit, and an Earth return phase to carry the sample back to Earth. The three phases would likely be launched from Earth in at least two separate launches in order to minimize the mass of each launch.[1] The MAV will be required to endure a long mission from Earth launch to Mars ascent, possibly as long as two years, including a stay of up to 90 sols on the Martian surface. The long duration in space and the high and low temperature extremes of the Martian surface would be challenging, and would require significant thermal protection for typical liquid spacecraft propellants. Therefore, under many current design proposals, the MAV would be a solid-fuel spacecraft in order to cope with the temperature extremes.

Solid fuel vehicles have several advantages. They are relatively stable for long term storage, and start up more reliably than liquid fuel systems following long duration exposure to temperature and pressure extremes. Solid fuel systems suitable for surviving the long duration and extreme environment exist; several have significant space heritage. Comparable liquid fuel systems, which could be expected to survive the long mission and harsh environment, are still in development and have much less, if any, space heritage.[1] The primary disadvantage of a solid

fuel vehicle is that it cannot be turned on and off, and must fire continuously from ignition until all fuel is expended. Solid fuel vehicles or vehicle stages are typically used for shorter operational duration, typically measured in minutes instead of hours or even weeks or years for some liquid-fuel systems. The current budget challenges that the U.S. space program is experiencing mean that if humans want to explore Mars, the moon, and other planets, it will likely need to be with automated spacecraft, not manned. Mars is a minimum of 4.3 light-minutes from Earth, which means a minimum of 8.6 minutes round trip for a signal from Mars to Earth and back to Mars. For almost any spacecraft, this is far too long to rely on Earth-based decision making in the event of a critical component failure. The mission could be a complete loss even before mission control knows there is a problem. Therefore, a system must be in place for autonomous onboard failure detection and isolation (FDI) and failure mitigation. At the very least, such a system could detect and mitigate the problem until mission control can issue new commands. Even for a spacecraft in Earth or Moon orbit, autonomous real time failure detection and mitigation provides significant benefits.

This thesis details algorithms for autonomous, real time propulsion failure detection, isolation and mitigation. The mitigation algorithms are designed for solid fuel spacecraft, where thrusters cannot simply be shut off when they fail. Also, the only sensor these algorithms require is an Inertial Measurement Unit (IMU), which is already present on nearly any spacecraft; they require no additional hardware modifications. Hardware modifications and hardware redundancy can be very expensive, and adding these algorithms provides increased redundancy with minimal extra cost.

Another advantage is that these algorithms are simple enough for real time operation in processing-constrained space systems. This is particularly relevant for a long duration and

performance-critical system such as the MAV, where any computer system must be durable and radiation hard, and therefore computing power is limited. Cosmic radiation that exists naturally in space can cause upsets and faults in silicon-based computer systems. Most vulnerable are memory devices, where a radiation particle can upset individual memory bits and swap a one for a zero, or vice versa, or even damage the circuitry permanently. Most systems could survive at least one upset, however multiple upsets could quickly degrade computing performance. Space-qualified computer systems must be hardened against radiation damage through a variety of methods. If a vehicle system requires a more radiation resistant computer, then the original component technology on which the computer is based will be older, and therefore it will be slower and have less memory capacity. A mission such as MSR and the MAV will require significant radiation protection, and because of this will have severely limited memory storage and computing capability, of which failure detection and mitigation will only be allowed a small portion. Limiting the computing and storage load of the algorithms described in this thesis provides significant benefits to the mission design and allows them to be implemented with minimal extra cost or time.

1.2 Vehicle Description

For the MAV, a solid fuel lifting stage is not unlikely in any design, and parallels many small and medium payload Earth launch lifting vehicles, such as the Orbital Sciences' Pegasus and Minotaur launch vehicles, and NASA's Scout Rocket. In the current design, the second stage of the MAV would be a solid rocket booster with a small liquid reaction control system (RCS) for attitude control. One topic that may be of interest is to use solid instead of liquid fuel for the RCS, in order to increase reliability and temperature survivability; however, this poses additional challenges. Solid fuel systems cannot be throttled on and off like liquid propellant

systems. One common method for solid fuel RCS is to use solid fuel gas generators, where the gas is released through valves at various points around the vehicle in order to exert desired forces and torques on the vehicle. An example of this is the Orion Launch Abort System Attitude Control Motor [2] developed for NASA's Orion program.

A particular challenge for solid-fuel RCS using a gas generator is that the system pressure must be regulated. The gases must be directed from the gas generator, through the vehicle and out to the valves. The vehicle could explode from over-pressure, and too low pressure would result in loss of vehicle control authority from decreased thrust. Solid fuel RCS can experience thruster failures just like liquid fuel systems, however the inability to halt solid fuel combustion on command makes dealing with the thruster failure more difficult.[3] On a liquid-fuel system, the fuel supply to a stuck jet can simply be stopped, ending the failure. Liquid fuel systems also require pressure control; however, this is usually implemented in the fuel tanks and is independent of jet firings. Therefore pressure control has much less impact on maintaining vehicle control authority and the mitigation of a jet failure in a liquid fuel RCS than it does in a solid fuel RCS. Both of these issues make failure detection and mitigation more difficult on a solid-fuel vehicle than a liquid-fuel vehicle.

While slightly different from the current MAV second stage, the hypothetical test vehicle for this research was chosen to show the validity of the research algorithms and maintain applicability to a wide of vehicles in six degree of freedom applications. The hypothetical vehicle has sixteen jets in eight opposed pairs, capable of controlling motion in any degree of freedom. The vehicle's mass properties are assumed to be well known. The detection algorithm and vehicle mass property estimates could be updated with any change in mass properties from known sources. Mass property changes from fuel depletion are well characterized, and solid fuel

means no disturbances from fluid motion. For the Mars Sample Return mission, the sample mass properties will not be known exactly, however the vehicle could use methods such as in [4] and [5], or the mass properties could be measured prior to launch by the rover system.

The environment of operation will be assumed to be free space with minimal disturbances acting on the vehicle. It will be assumed that a previous boost stage placed the vehicle on the necessary trajectory for Mars orbit. The system considered in this test case will be responsible for fine-tuning the orbit and maneuvering the sample for docking with the Earth return stage. The following assumptions will be made about the vehicle failure modes. First, only one jet failure will occur at any one time; multiple simultaneous failures are outside the scope of this thesis. Also, the two failure modes considered will be valves stuck on and stuck off. Stuck on represents that the valve is firing at 100% thrust at all times while failed. Stuck off means that the valve does not fire when commanded, i.e. it has a thrust level of 0% at all times. These will also be referred to as on and off failures respectively. The terms valve, thruster and jet will be used interchangeably in this thesis to represent the valves which expel gas to exert force on the vehicle. Also, as the failures are considered to occur exclusively when a valve has stuck in one of two orientations, the terms stuck valve and jet or thruster failure will be used with the same meaning.

Two additional failure modes to be aware of are partial on failures and off-nominal thrust. Partial on failure is when a valve is stuck on, but firing at less than 100% and more than 0% thrust at all times, regardless of command. Off-nominal thrust may be thought of as a partial off failure, where a valve fires at less than 100% thrust when it is commanded, but still fires only when commanded. The reader should be aware of these failure modes, however they will not be

fully discussed in this work and a complete detection and mitigation scheme has not been developed for these failures.

1.3 Relevance and Thesis Outline

This research is primarily focused on space systems, however it is applicable to a wide range of research areas. The failure detection method detailed here could be applied to nearly any vehicle. The basic concept is easily transferable, comparing expected with actual motion and determining the disturbance experienced by the vehicle. This disturbance signature can then be compared to a set of expected failure modes to determine the likelihood of each particular failure. The mitigation strategy could also be applied to vehicles with non-opposing thrusters. This would require significant controllability analysis for each vehicle and all possible thruster failures and combinations, but it would be possible to predefine which opposing thruster(s) should be fired if a particular thruster suffered an on failure. A slightly different application of these algorithms could also be used to determine thrust levels for a vehicle, in order to improve control efficiency and accuracy. [5]

Chapter 2 explains the design and options for the chosen fault tolerant controller on which these detection and mitigation algorithms are based. Chapter 3 of this thesis details the Failure Detection and Isolation Algorithm, including available algorithm options, the chosen algorithm, and methods for simultaneously detecting either a stuck off or stuck on valve failure. Chapter 4 explains the Failure Mitigation Algorithms, including valve isolation, disturbance mitigation, and updating the jet selection algorithms to accommodate the stuck valve. Chapter 5 contains the simulation description and research results. Chapter 6 is the Conclusion and also details options for future work.

2.0 Control Law and Jet Selection Algorithm

Selection of the vehicle autopilot controller is crucial to the development of detection and mitigation algorithms. These control elements must work together to create a robust and well controlled, high performing vehicle. Throughout the course of the research associated with this thesis, numerous problems were encountered when the autopilot controller was conflicting with elements of the detection or mitigation algorithms. For optimal performance it is necessary that the autopilot controller should also be fault tolerant and have some built in robustness to disturbance. One type of typical spacecraft autopilot is divided into two pieces, the control law and the jet selection algorithm. The two elements of the control system are highly coupled, and unless commands are executed in the correct order and the correct signals are communicated between the two algorithms, the entire system can perform in undesirable and unpredictable ways. Therefore it is necessary to discuss the vehicle control law and jet selection before detailing the failure identification and mitigation algorithms which are the focus of this thesis.

2.1 Phase Space Vehicle Control Law

For this research, a phase space method has been selected for use as the control law, based on [5], [6] and [7]. This particular phase space control law has been shown to be effective at controlling vehicle attitude and linear velocity for a spacecraft using RCS control. The primary elements of the control law are: the velocity to be gained vector, the multi-dimensional phase space, and the state deadband.

2.1.1 Velocity to be Gained

The velocity to be gained vector is essentially a desired rate change vector calculated from the current state and the desired state. For a vehicle with a state vector $\underline{x}$, which has a

derivative $\underline{\dot{x}}$, the vehicle desired state is $\underline{x_d}$ and the desired rate is $\underline{\dot{x}_d}$. An augmented rate change vector, called the velocity to be gained, is defined as:

$$\underline{\dot{W}} = c \cdot unit(\underline{x_d} - \underline{x}) + (\underline{\dot{x}_d} - \underline{\dot{x}}) \tag{2-1}$$

The component $unit(\underline{x_d} - \underline{x})$ is the unit magnitude direction which drives the vehicle from the current state to the desired state. The scalar value c is called the convergent velocity, and $(\underline{\dot{x}_d} - \underline{\dot{x}})$ defines the desired change in vehicle rate. It can be seen that when the vehicle rate equals the desired rate, the only correction is to vehicle position, and when the vehicle position equals the desired position, the only change is to vehicle rate. In this way both position and velocity can be accurately controlled in any dimension. Because errors exist in any system, the relative position and velocity to the target cannot be precisely known or controlled. It is also not possible to effect the rate change instantaneously. Both of these limitations mean that frequent calculation of velocity to be gained will result in frequently changing requests. Therefore it is necessary to incorporate deadbands into the control law in order to produce a reasonable and feasible controller.

2.1.2 Phase Space Deadband Concept

Consider a vehicle described as above, with a new vector $\underline{x_e}$, defined as:

$$\underline{x_e} = \underline{x} - \underline{x_d} \tag{2-2}$$

where the vector has j dimensions, as many as the vehicle has states. It is necessary to control each state in order to keep the each within its defined deadband. The deadband db_i for each state is the region of space within which the state is acceptably close to the desired value. Assuming that the deadband for each state is symmetrical, that is that dbi+ equals dbi-, then to determine if the vehicle is within its deadband, it is necessary to compare only $\underline{x_{ei}}$ and db_i. If $\underline{x_{ei}}$ is greater

than db_i, the state is outside its deadband. The space described by the j state deadbands can be visualized as a j-dimensional prism. In order to equalize vehicle control in disparate dimensions, such as when the deadband for linear control may be much greater numerically than that for rotation, it is reasonable to normalize the deadbands. The normalized j-dimensional prism becomes a hypercube, within which a hypersphere, a sphere of greater than three dimensions, can be inscribed. The easiest test for whether the vehicle has exceeded its normalized deadband is to compare two scalar values:

$$\sqrt{\underline{\tilde{x}}_e \cdot \underline{\tilde{x}}_e} < r_{ps} \tag{2-3}$$

where $\underline{\tilde{x}}_e$ is the normalized state vector, and r_{ps} is the radius of the hypersphere. This hypersphere shall be called the "phase sphere."

2.1.3 Phase Space Control Law

The phase space control law is the combination of the velocity to be gained vector and the phase sphere concept into one simple, robust control law. As described above, it is necessary to normalize the individual state deadbands db_i. This is primarily because they may have significantly different numerical values. The deadband for control of translation may be 10 meters, however the deadband for rotational control may be only 0.5 degree. Calculating the convergent velocity along $(\underline{x}_d - \underline{x})$ only would direct the vehicle along the numerically largest error, not along the state which is furthest outside its deadband. That is, the controller would not focus on the state which most needs to be corrected and may seriously over control certain states which happen to be measured in units which result in greater numerical values. Typically this would result in over controlling translation at the expense of rotation. Therefore the phase space control law uses an augmented velocity to be gained principle which directs the convergent velocity along $\underline{\tilde{x}}_e$:

$$\underline{\dot{W}} = -c \cdot unit(\underline{\tilde{x}}_e) + (\underline{\dot{x}}_d - \underline{\dot{x}}) \tag{2-4}$$

where c is determined based upon where $\underline{\tilde{x}}_e$ is relative to the state deadbands.

There are several characteristics of the phase space control law which are worthy of discussion. First, the control law displays limit cycling behavior. Limit cycling is defined as the oscillation of a system about its desired state, due to an inability to precisely control the system states to the desired value without relative rate. The primary cause of limit cycle behavior is that thrusters have a finite minimum impulse which they can produce, and therefore once the vehicle has been driven into the deadband, it is not possible to precisely drive the relative rate to zero. Therefore, even in the nominal case, limit cycles will occur. Limit cycling can also result from outside disturbances and control or feedback imperfections. As a vehicle is disturbed away from its desired state, control will be applied to return it to the desired value. It may overshoot the state, or disturbances may push it past the desired state, and therefore more control will have to be applied to return to the desired state. This is limit cycling. If a vehicle achieves its target state, but still has a relative rate for that state, then it will likely limit cycle about that desired state.

As a position state reaches its deadband with a nonzero rate, the rate must then be reversed to push the state back towards its desired value. Because it is impossible to instantaneously change the rate when the state reaches the edge of the deadband, it is necessary to define a second phase sphere within the first. The size of the inner phase sphere is set so that any state variable may be corrected after passing this sphere, but before exceeding the outer deadband. The magnitude of the inner phase sphere will likely be different in each dimension, even in normalized space, because the control modes and level of control authority will often be different for each dimension and its associated effectors. The entire hyper-dimensional space

can be divided into three regions based on the two phase spheres. If the vehicle states are within the inner sphere, the vehicle is in region 1. If the states are within the outer sphere but outside the inner sphere, that is region 2, and outside the outer sphere is region 3.

The final consideration is to set the value of c based on the vehicle's location in the phase space regions. If the vehicle is in region 3, then c will be set reasonably high to achieve an acceptably fast convergence rate. If the vehicle is in region 1 or 2, then c will be set to a reasonable limit cycle rate. Too slow of a limit cycle rate may be difficult for the vehicle to achieve, such as if it is below the minimum impulse achievable by a jet, and too fast of a limit cycle will result in extraneous jet firings and unnecessarily frequent vehicle departures from regions 1 and 2. When coming from region 3, a braking rate change is applied to counteract the tendency of the vehicle to overshoot the target and reenter regions 2 or 3. Figure 2-1 shows a convergence trajectory in a two dimensional phase space, and Figure 2-2 shows a limit cycling behavior in the same two dimensional space. Points a, b, c, and d represent successive points in time order.

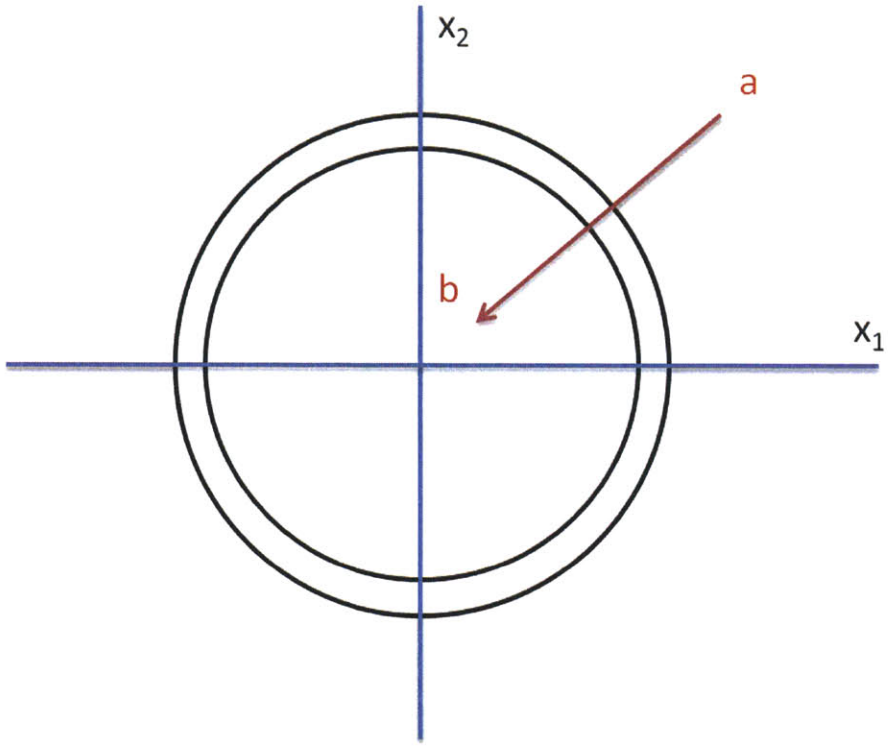


Figure 2-1: Convergence Trajectory

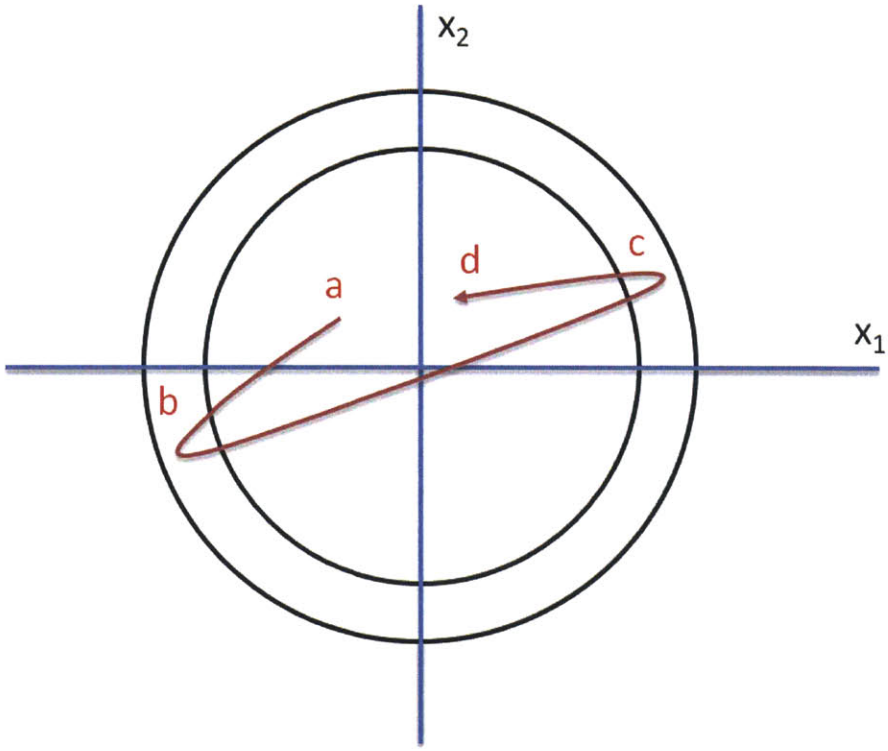


Figure 2-2: Two-Sided Limit Cycle

2.1.4 One-Sided Limit Cycle

The final noteworthy characteristic of the phase space controller is its limit cycling behavior in the presence of a disturbance acceleration. A typical limit cycle is two-sided, meaning that as a state is corrected back towards its desired value and overshoots, it will then likely exceed its dimension on the inner phase sphere on the side opposite its previous excursion. When a disturbance is acting on a vehicle state, that state will display one-sided limit cycling behavior, shown in Figure 2-3. β represents the disturbance acceleration, whether from a failed jet or external source.

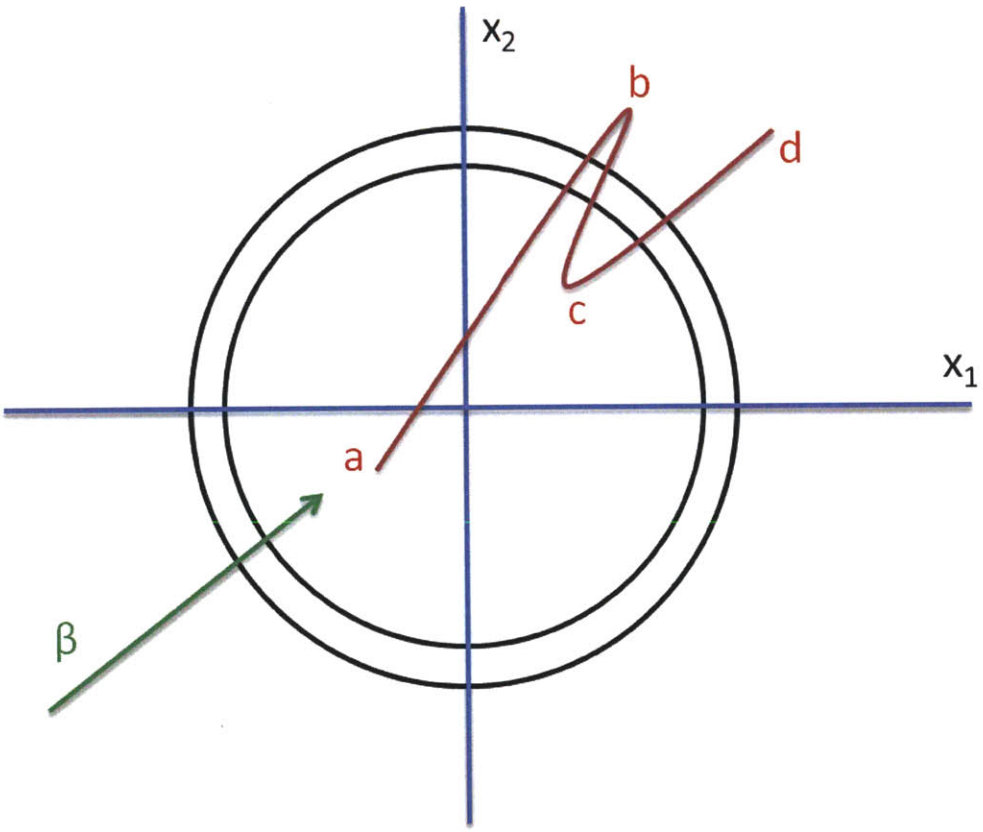


Figure 2-3: One-Sided Limit Cycle

The one-sided limit cycle behavior occurs because as the vehicle exerts temporary control to drive the state from region 2 back into region 1, the disturbance will again push the state back

towards region 2 as soon as control is reduced inside region 1. Over time the axis of the limit cycle will move so that it is oscillating nearly directly along the direction of the disturbance.

These principles can easily be applied to spacecraft control, where $\underline{x}$ and $\underline{x}_d$ can be defined to contain whichever vehicle states are to be controlled. Most commonly this will be the six dimensions of angular and linear position, as is the case in this research. Whenever the vehicle is in regions 2 or 3, or enters region 2 from region 1, a new rate change request is generated and a command is sent to the jet selection algorithm to generate a new set of jet commands and firing times to meet the rate change request. In order to stabilize the control and avoid excessive control oscillations, even if the vehicle remains in regions 2 or 3, a new rate change request and associate jet selection will not be made until the previous jet selection has been fully executed, that is, until the jets have stopped firing.

2.2 Jet Selection and Linear Programming

One of the most powerful and capable methods for jet selection in a multi-jet vehicle is linear programming. As shown in [8], jet selection for a multi-jet system can be expressed and solved as a linear programming problem, and it is useful to begin with this method in order to explain the nature of the problem.

2.2.1 Linear Programming Formulation

The generic linear programming problem has three characteristics: a linear objective function which is to be minimized or maximized, a set of linear constraints which must be met (typically equalities or inequalities), and all variables are bounded to be non-negative. This formulation lends itself well to the selection of jets for a multi-jet spacecraft. The objective function to be maximized or minimized is defined as:

$$f(x) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{2-5}$$

Where c_i is a constant weight, such as the rate of fuel expended by firing a jet, and x_i is the selection variable, such as the jet firing time. The objective function is also referred to as the cost function, where all c_i can be visualized as the cost of each x_i. This function is subject to the constraints

$$A\underline{x} = \underline{w} \tag{2-6}$$

and

$$x_i \geq 0, \text{ for all } i \tag{2-7}$$

where Equation (2-6) is also referred to as the equality constraint, and Equation (2-7) is an inequality constraint.

The set of solutions to a linear programming problem, that is, all the possible values of x which satisfy the constraints, form a convex set. Any set of equalities or inequalities such as Equations (2-6) and (2-7) will only create a convex set. A convex set is defined as a set such that a segment joining any two points in the set is also in the set. A simple example of this can be shown in two dimensions. Figure 2-4 shows a convex set, and Figure 2-5 shows a non-convex set.

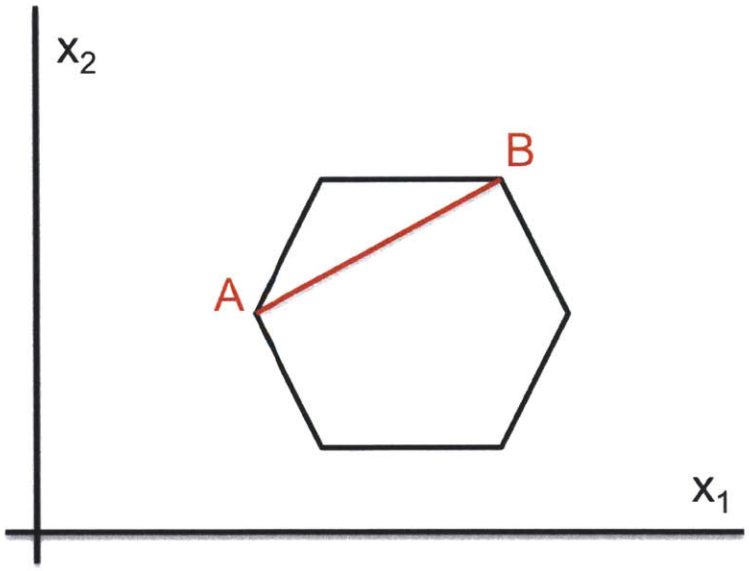


Figure 2-4: Convex Set

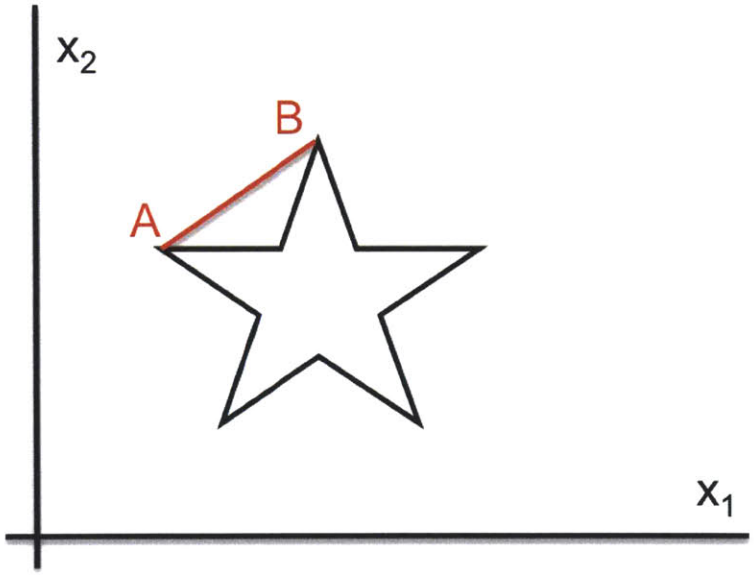


Figure 2-5: Non-Convex Set

It is possible to find two points such as A and B in Figure 2-5 for which the segment between them is not completely within the set. There are no such two points in Figure 2-4.

An optimal solution to the linear programming problem will be found at an extreme point on the set. An extreme point is any point which is not on a segment between any two other

points. Points A and B in Figure 2-4 are both extreme points. Point A is the point which minimizes x_1 and Point B is the point that maximizes x_2. Changing the values of any variable at an extreme point changes the value of the objective function. The optimal extremum is that point which has the most optimal value of the objective function. Other points within the set may result in the same value of $f(\underline{x})$ as the optimal extremum, however at no point will the cost function have a more optimal value. Therefore, searching only the extreme points of a set gives the linear programming solver a finite number of solutions to check. It is obvious that, as a problem expands beyond two dimensions, the number of extreme points grows prohibitively large and becomes time or computationally intensive to check all extrema individually. Therefore it is beneficial to use algorithms which limit the number of points to check.

2.2.2 The Simplex Method

The Simplex method is one such method of linear programming which limits the number of points which are to be checked. The basic concept is that the Simplex method proceeds from one extremum only to other extrema that improve the objective function, that is, to extrema with more optimal (higher or lower) cost. In order to explain the Simplex algorithm, it is necessary to first define several terms.

A solution is any vector $\underline{x}$ which satisfies the equality constraints of Equation (2-6). If it also satisfies the inequality constraints of Equation (2-7), it is a feasible solution. A basic feasible solution is a feasible solution $\underline{c}$ which has only as many nonzero elements as equality constraints imposed on the problem. In jet selection, this means that $\underline{c}$ has only as many nonzero elements as degrees of freedom which are to be controlled. If the activity vectors $\underline{x_i}$ which correspond to these nonzero elements of $\underline{c}$ span the selection space, then they form a basis of a

finite dimensional vector space. This means that any vector $\underline{x}$ can be represented as a linear combination of the basis vectors $\underline{x_i}$.

The Simplex method starts with any basic feasible solution and seeks to replace vectors in the basis with other vectors which are not currently in the basis. The method improves cost by adding vectors to the basis, but must maintain the solution feasibility when removing any current basis vectors. The steps are repeated until an optimal solution is found, or the algorithm determines that no other feasible solutions exist, i.e. the current solution is the best possible solution. It is necessary to define the two major rules of Simplex, which decide how vectors are eliminated from the basis, or excluded, and also how they are added into the basis, or included.

In Simplex, the solution is always maintained as a basic feasible solution, and therefore for every vector which is included, another must be excluded. The first rule is to obtain a new basis and a new feasible solution by including $\underline{x_k}$ into the basis in the place of some vector $b_{(i)}$ currently in the basis, as long as $b_{(i)}$ is selected to maintain the feasibility of the solution. This is determined by adding a slack vector to the basis, and then determining the first basis vector $b_{(i)}$ which can be set to zero to maintain the feasibility of the solution.

The second rule details how a vector should be included into the basis. Let n be the number of elements in $\underline{x}$, and q be the number of inequalities in Equation (2-6) and $\underline{x_j}$ is the new vector which is to be tested for inclusion in the basis with corresponding cost c_j. By adding the new vector, Equation (2-5) can be updated as follows:

$$f(\underline{x})' = \left\{\sum_{i=1}^{q-1} c_{(i)} b_{(i)}\right\} + c_j x_j \tag{2-8}$$

The first term is for $i = 1$ to $q - 1$ because in the previous step one basis vector $b_{(i)}$ has been eliminated from the basis. If the objective function $f(\underline{x})$ is to be minimized, then the first vector

x_j which decreases $f(\underline{x})$ is subsequently included in the basis. If the objective function $f(\underline{x})$ is to be maximized, then the first vector x_j which increases $f(\underline{x})$ is subsequently included in the basis. The process continues until no vector can be found which will either increase or decrease $f(\underline{x})$, whichever is desired, beyond the current value. This method can also be visualized as beginning with the basic feasible solution, progressing to the first available extremum, and then moving along an edge of the set from the first extremum to another extremum of more optimal cost.

2.2.3 Applying Linear Programming to Jet Selection

As previously mentioned, jet selection for a multi-jet system can be solved as a linear programming problem. This section details a common application of linear programming to jet selection.

Paired with the chosen phase space control law, the purpose of the jet selection is to achieve the desired rate change $\underline{W}$ with the available selection of jets. In the linear programming problem, the jets are represented by their activity vectors, which make up the basis. The jet activity vector represents the acceleration achieved by firing jet j. In implementation this can also be represented as the change in angular and translational body rates achieved by firing jet j for unit time. The six-element activity vector α_j is given by the following equation:

$$\alpha_j = \begin{bmatrix} I^{-1} \cdot (r_j \; x \; T_j) \\ \cdots \\ T_j/m \end{bmatrix} \tag{2-9}$$

where r_j is the displacement from the vehicle center of mass to the jet and T_j is the jet thrust vector. The vehicle mass is represented by m and I^{-1} is the inverse of the vehicle's three-

dimensional inertia matrix. The first three elements of the activity vector represent rotational acceleration and the last three elements represent linear or translational acceleration. A key assumption here is that the second-order terms of α_j are negligibly small. Because the state measurements and estimates provided by the IMU are assumed to be very good, especially over short periods, this is a logical and acceptable approximation.

Having defined the selection basis, it is next necessary to define the objective function which is to be optimized. The following sections discuss likely options for objective functions, associated challenges and alternate linear programming formulations in greater detail. A common objective is to minimize fuel usage, for which the objective function $f(\underline{x})$ is defined as:

$$f(\underline{x}) = \sum_{i=1}^{\#jets} x_i f_i \tag{2-10}$$

where x_i is the jet firing time and f_i is the jet's rate of fuel consumption. Because no jet can be fired for a negative time, the corresponding inequality constraint is:

$$x_i \geq 0 \text{ for all } i \tag{2-11}$$

In order to execute the desired rate change request, the equality constraints for the optimization problem are:

$$\underline{W} = \sum_{i=1}^{\#jets} \alpha_i x_i \tag{2-12}$$

A realistically solvable problem requires three things. First, the cost function must be linear and have a minimum, which makes the problem achievable and ensures that the cost function is useful in a linear programming formulation. Second, the equality constraints must allow for one or several solutions, ensuring that the desired rate change request can be achieved

with the given effectors (jets). Finally, the inequality constraints must be met by one or more of these solutions, where the goal is to achieve the desired rate change request in a reasonable time.

2.2.4 Time and Fuel Optimal Formulation

There are several possible formulations of the objective function $f(\underline{x})$ which may be used with the linear programming solver and optimization. The most common is typically minimum fuel optimization, especially in liquid fuel vehicles. In a liquid fuel system, minimizing fuel usage increases the achievable mission lifetime. For a solid fuel system such as the MAV, fuel consumption cannot be controlled to a level which would significantly contribute to mission lifetime. Simulation has shown, however, that the minimum fuel formulation works acceptably well for this research, and will be detailed in Chapter 5.

An alternative formulation is to minimize the vehicle reaction time, e.g. to minimize the maximum of all the jet firing times. For the minimum time solution, the objective function $f(\underline{x})$ is defined as:

$$f(\underline{x}) = \max(x_i) \tag{2-13}$$

Therefore the linear programming solver seeks to find the solution which accomplishes the desired rate change request in the minimum time. In most cases, this will be the solution which fires the greatest number of jets simultaneously. In many cases this may be desirable, such as when a vehicle is maneuvering rapidly, however this may not always be true. There may be times, for instance when performing delicate or fine maneuvers such as docking or payload release, it is not desirable to have a high number of jets firing at the same time. Therefore, this formulation likely should not be used at all times for a spacecraft autopilot.

In certain modes of operation, however, it may be desirable to accomplish the minimum time maneuver, or to optimize for a combination of time, fuel or another criteria. An example combined objective function would be:

$$f(\underline{x}) = \left\{ \sum_{i=1}^{\#jets} x_i f_i \right\} + g \cdot \max(x_i) \tag{2-14}$$

Where g is a weighting coefficient which can be set as appropriate for the application. Obviously a higher value of g would increase the weighting on the minimum time element and cause the solver to more heavily focus on minimizing the vehicle reaction time. A lower value of g would cause the fuel minimization element of the problem to take precedence.

Another possible formulation would be to make the objective function simply a linear combination of individual variables x_i, such as:

$$f(\underline{x}) = g \cdot x_1 + h \cdot x_2 + k \cdot x_4 \tag{2-15}$$

where g, h and k are weighting coefficients. This problem can be understood as an adaptation or simplification of the above minimum fuel optimization, where g, h and k would be considered as the associated fuel burn rates for jets 1, 2 and 4. It is possible to conceive an example where it may be desirable only to optimize the fuel used by a specific set of jets, such as if they used an expensive or scarce fuel. Just as above, increasing the value of any coefficient relative to the others will increase that term's weight in the objective function, and therefore cause the linear programming solver to optimize the associated variable more heavily.

Such decisions on weighting of disparate elements or combining different types of objectives in the objective function require a thorough analysis of vehicle dynamics and the impacts of the different linear programming formulations on autopilot behavior. Therefore

optimization combinations will not be considered for this research, but is useful for discussion and understanding the function of the linear programming solver.

2.2.5 Regular and Upper Bound Simplex

In choosing a linear programming solver it is important to understand both the capabilities and limitations of the available solvers. The regular Simplex method as described above is very efficient, however it can only select as many jets as degrees of freedom which are to be controlled. In order to understand this limitation, consider a brief example where the Simplex method is being used to control motion for the hypothetical MAV, but only in three degrees of freedom. As in Figure 2-6, if the desired rate change is in the +X direction only, there are four jets which exert force and cause acceleration in the +X direction. The time optimal solution would be to fire all four jets simultaneously, exerting the most amount of force over the shortest time. Regular Simplex, however, would only select a maximum of three jets (the same number as the degrees of freedom it is controlling), and possibly only two jets if the third jet would create conflicting torques on the vehicle.

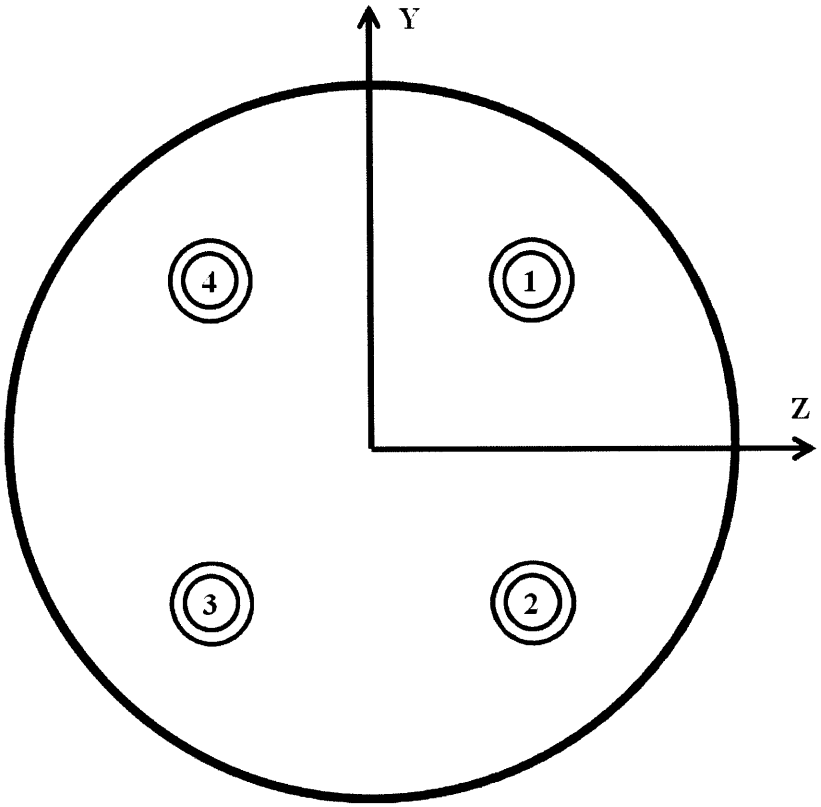


Figure 2-6

An altered version of the Simplex method, called Upper Bound Simplex, is capable of selecting more jets than degrees of freedom. In this formulation there is also a maximum value, an upper bound, that the selection variables (the jet firing times x_j) may not exceed. This is an additional inequality constraint on the problem:

$$x_i \leq x_{\max}, \text{ for all } i \tag{2-16}$$

The primary means of altering Simplex to include the upper bound condition is to change the way variables are incorporated into and excluded from the basis. This is done by considering both whether the proposed variable change improves the objective function, and what the current value of the variable is relative to its upper and lower boundaries. Another method is detailed by

Crawford in [8], which shows how the regular Simplex method may be adapted to find the time optimal solution using slack variables and a magnified rate change request.

2.2.6 Chosen Linear Programming Formulation

The primary difference between the optimization options and versions of the Simplex method would be in which jets are fired, how often and for how long. As any formulation must satisfy the equality constraint, such as Equation (2-6), to match the desired rate change request vector, every formulation should create the same end result within a reasonable amount of time. The greatest difference will come in the system pressure, which changes significantly based on how many jets are fired. Pressure control is not considered in this thesis, and algorithm adaptations or additional control elements would be required in order to control pressure. Therefore it is not possible to fully analyze the impact of the different linear programming solvers on the final vehicle. A reasonable choice can be made, however, in order to enable the stuck valve detection and mitigation work which is the focus of this thesis.

The version of the Simplex method that will be used for this research is regular Simplex solving the optimal fuel problem. The primary reason for choosing the fuel optimal formulation is that is tends to fire a minimum of jets. The foundation of the detection algorithm is to predict the expected change in vehicle motion, measure the actual change, and if the difference is great enough it can be attributed to a failure of one or more jets. Firing fewer jets increases the probability of success for the detection and mitigation algorithms, because the fewer jets that are fired, the less often the motion prediction will change. Firing fewer jets also means that vehicle rates will likely oscillate less and change less rapidly, which decreases the chances that higher-order vehicle dynamics will become significant and distort the predicted motion. Also, as will be discussed in the mitigation section, a stuck on failure will deplete pressure very rapidly, and

firing an opposing jet will decrease pressure and therefore vehicle control authority even more quickly. Solving for the minimum fuel solution would likely help to decrease the impact of this in the final vehicle, and is therefore a likely choice when implementing a solution that would include pressure control.

2.3 Jet Selection Algorithm Options and Previous Research

The following options were also considered for use as the jet selection algorithm. First, a method could be implemented similar to the Space Shuttle On-Orbit Autopilot, which is detailed in [9] and [10]. The Shuttle jet selection algorithm for the primary RCS jets is specifically tailored to the Shuttle vehicle, and is based on lookup tables and extensive thought experiments to determine which jets shall be fired to accomplish specific maneuvers or rate changes. As designed and implemented the jet selection tables are two-fault tolerant, able to maintain control of the vehicle in the presence of any two jet failures. There is some redundancy beyond two faults, however there are cases of three faults in which the vehicle would be unable to control motion in all six degrees of freedom. Also, the number of lookup tables required to maintain controllability for three or more failures becomes significantly unwieldy. The original space shuttle had a redundant set of five flight computers, which were not capable of carrying all necessary data for the entire mission. At launch, four computers carried launch data, and one would carry the abort data to return the shuttle and crew safely to the surface if a problem occurred in transit to orbit. Throughout the mission, successive computer modules such as on-orbit control and reentry had to be loaded one at a time, as they were needed. Given these computational and memory storage limitations, the table lookup approach as detailed above was necessary and also very successful. This approach would, however require a prohibitive amount

of effort to replicate for another vehicle. Therefore an alternative method for jet selection will be used in this research.

Another method is as published by Glandorf in "An Innovative Approach to the Solution of a Family of Linear Programming Problems" [11]. The family of linear programming problems that Glandorf is referring to includes the problem of jet selection for a six degree of freedom multi-jet spacecraft, again as shown by [8]. The key insight is to separate the calculation of rate change vectors, lists of jets to fire, and inverse basis matrices (arranged into sets of basis solutions) from the selection of the optimal solution for the given rate change request. In this way, basis solutions can be pre-calculated for a given set of vehicle configurations. These solutions can be stored in memory, and thereby decrease the real-time computation load on a vehicle's onboard processor, because the processor must only select from a given set of options instead of performing the entire linear programming search.

If we briefly assume that the vehicle mass properties are constant, one set of solutions would be stored in memory for the nominal case, and one set would be stored for each failure mode for each jet; two failure modes per jet for sixteen jets makes thirty-two more solution sets, and thirty-three total. In a relatively stable case, this method provides an advantage over typical linear programming methods, and remains within reasonable memory limitations. As more complications are added, however, the memory requirements increase significantly and the benefit decreases. Changing mass properties, for instance, require separate basis solutions, and so do extra failure modes. This method also does not take advantage of the computational benefit of opposed jet pairs, as [7] does. Therefore the method in [11] will not be used in this research.

Reference [7], "Precise Nulling of Attitude and Motion Errors of a Spacecraft Using a Phase Space Autopilot" by Kellogg, details a computationally simple method for calculating on times for a defined set of jets using the pseudo inverse technique, referring to a method of matrix manipulation for non-square matrices. Kellogg showed that the pseudo inverse jet selection worked acceptably well only for vehicles with opposed pairs of jets. The reason is that the pseudo-inverse is primarily just a matrix manipulation, and therefore can produce negative firing times. In a vehicle with directly opposed jets, a negative firing time for one jet can simply be inverted and added to its opposing jet. Also, even with directly opposed pairs, the pseudo inverse method can result in a singular matrix, meaning that the desired vehicle rate change request cannot be achieved with the available jets. This approach would work for the MAV as described, but would not work for a different vehicle without directly opposed thrusters. Also, once a jet fails, there is no longer a complete set of directly opposed thrusters. When this occurs some of the necessary matrix manipulations lose full rank, and which increases the likelihood of creating a singular matrix or an unsolvable request. Therefore the pseudo-inverse jet selection method will not be used here because it limits the extension of this work beyond one hypothetical test case and might over-constrain the failure mitigation algorithm.

2.4 Fault Tolerant Autopilot Controller

The combination of Simplex with the phase space control law was test-flown successfully on the Space Shuttle, and is commonly referred to as the Orbital Experiment (OEX) Digital Autopilot (DAP).[12] This method has several advantages over those examined above. Simplex does not require predefined sets of jets and failure modes, as is necessary in [11]. On most modern space-qualified computers, common implementations of the Simplex method can be expected to perform all calculations in flight. Linear programming in general and Simplex as a subset of that can accommodate a wider range of possible configurations than the pseudo-inverse method. Simplex can handle in-flight, real time changes in jet thrust levels and vehicle mass properties without any changes to the problem formulation; the variables need only be updated as available. Within a reasonable level of expectation, Simplex will calculate the necessary jets to fire in order to achieve the desired rate change, if it is possible. The phase space autopilot, as detailed above, is also very robust to disturbances and as long as the vehicle thrusters are capable of counteracting the disturbance, the vehicle will simply limit cycle within its deadband and will not lose control. This autopilot configuration provides an optimal combination of robustness and flexibility without excess real time computation load or memory burden. Therefore, this approach will be used for the hypothetical MAV in this research.

3.0 Failure Detection and Isolation Algorithm

3.1 Algorithm Heritage and Options

This work is based on previous research conducted at the Charles Stark Draper Laboratory and elsewhere into detecting spacecraft thruster failures. Three previous works in particular will be discussed here. First is Reference [13] "Mass Property Estimation with Jet Failure Identification for Control of Asymmetrical Satellites" by Levy. Levy's work begins with estimating mass properties of a vehicle using an adaptation of the Extended Kalman Filter. As an extension of his work he is able to estimate jet thrust levels using the residuals from the mass property estimations. The strengths and weaknesses of this approach are largely the same as those of the Kalman Filter. The strengths include that by nature it is relatively insensitive to noise perturbation, and that the Kalman filter provides the optimal linear estimate of the vehicle states. The greatest weakness is that it is computationally intensive, and would likely be achievable but burdensome on a vehicle such as the MAV. Implementation for this case would be a slightly simpler version of Levy's work, because the mass properties are assumed to be well known. For failure detection, only the residual jet thrust estimation elements of the reference would be implemented. Levy's algorithm could also be used to estimate the mass properties of the MAV and the returning sample in real time in-flight.

The second is Reference [14] "Maximum Likelihood Failure Detection Techniques Applied to the Shuttle RCS Jets" by Deyst and Deckert. The authors use only the gimbal angle and linear velocity measurements available from the Space Shuttle orbiter IMU. The IMU measurements are incorporated into a steady-state constant covariance (fixed gain) Kalman filter to produce estimates of angular and linear disturbance acceleration. If the disturbance acceleration is great enough, the jet failure detection algorithm is triggered. The estimated

disturbance acceleration is then compared to the expected acceleration from each jet, which produces a likelihood estimate that each jet has failed. Assuming that only one jet has failed at a given moment, the jet with the greatest likelihood of failure exceeding a preset threshold is assumed to have failed.

Another relevant previous work is Reference [15] "Robust Failure Detection for Reentry Vehicle Attitude Control Systems" by Agustin, which details an advanced method for estimating RCS jet and aerosurface performance and potential failures during reentry of a reusable space vehicle, such as the space shuttle. Agustin's work used a robust two filter approach, with one filter tuned to detect thrust levels and failures in RCS jets, and another filter tuned to detect failures in aeronautical control surfaces. The need for two robust filters in this case arose from significant plant and model uncertainties given the complex, unpredictable and high-speed nature of reentry from space. In the case of the MAV, which for this report is assumed to be in free space with negligible atmosphere, such robustness and its associated complexity are not necessary. Therefore this method will not be used for the MAV discussed here.

One key point to note about [13] and [14] is that neither requires extra dedicated sensors for FDI. The method for [14] was proposed primarily for FDI on the Space Shuttle, which at the time in early stage design did not have dedicated sensors for FDI. The reason this algorithm was not chosen for incorporation in the final Space Shuttle design is that the Shuttle experienced a leak failure mode in which the thruster valves did not properly close when the thruster was turned off. The resulting leak produced thrust of about 11% of nominal, and the algorithm took over one minute to detect the failure. This was deemed to be too long, and instead the Shuttle designers added dedicated FDI sensors on the jets to detect the leak failures and also on and off failures, which [14] refers to as "hard failures." Leak failures were detected by temperature

sensors at the valve, when leaking propellant froze and chilled the injectors. Off failures were detected using chamber pressure sensors, and on failures were detected using both chamber pressure sensors and solenoid valve current.

This research assumes that the hypothetical MAV does not have dedicated FDI sensors in order to maintain design simplicity and decrease cost and mass. This is not uncommon in complex space missions, where any increase in mass is an automatic increase in cost. Therefore spacecraft are typically built with the best possible parts in the most careful way possible in order to achieve the maximum reliability with the minimum of hardware redundancy. For example the Apollo Guidance Computers were built with very carefully selected parts.[16] As a result, they were highly expensive, but no Apollo Guidance Computer ever failed in use. Such maximum quality and minimum redundancy is especially true of automated missions where no human life is in danger from a fault or mission failure. Decreased reliability can be exchanged for greater capability or lesser cost when the greatest danger is a loss of mission, not loss of life. Another benefit of this research is that an adapted version could be loaded onto another existing vehicle without additional hardware. This could be done for a large fleet of vehicles already in service, for which retro-fitting modifications is cost prohibitive, or for a vehicle in flight, such as a deployed space probe, where hardware modification is impossible or nearly so.

3.2 Possible Error Sources, Considerations and Assumptions

Before detailing the entire stuck valve detection algorithm, it is important to discuss possible sources or error, how they will be dealt with, and any assumptions which are made for this research. This section does not seek to exhaustively analyze all possible issues which could befall a MAV in a real MSR mission, only to show that this research could be effectively applied to a real vehicle.

The first possible source of system errors is measurement noise and measurement error in the MAV telemetry data, including attitude, position, angular rate, velocity, angular acceleration and linear acceleration, which are collectively referred to as the vehicle states. Measurement noise is any unwanted disturbance in the measurement signal which can come from a variety of sources. Noise varies with time and is not predictable exactly at any given time, however it can be characterized using primarily probabilistic measures, often based on the sensors and hardware to be used. In preparation for the MSR mission, significant testing and evaluation would be conducted on all elements and the final vehicle. Through this testing, data could be gathered and sensors and hardware could be analyzed to estimate noise characteristics. This data could also be combined with data and experiences from previous Mars missions to generate a best estimate of what the MAV measurement noise characteristics will be in flight. Based on the analyzed noise characteristics, a variety of effective filters could be designed to maintain the most accurate possible estimations of the vehicle states.

Measurement error sources will be characterized along with measurement noise during initial testing. One of the greatest contributors to measurement error is initialization errors, where the initial position and attitude of the MAV would be set incorrectly based on errors in the launch system. These errors could be minimized by increasing measurement accuracy and redundancy in the launch system, either through sensor redundancy or inclusion of different types of sensors, such as stellar sighting.

For the purposes of this research, it is assumed that the best possible filters and error countermeasures are incorporated into the vehicle IMU or support electronics, and that no further processing or noise considerations are necessary in the failure detection algorithm. The impact

of noise will be discussed in the simulation section, however the data provided to the failure detection algorithm will still be assumed to be the best possible estimate.

Another characteristic of the vehicle which must be taken into consideration is the fact that valves do not turn instantly on and off when commanded. Once the valves are open, it takes a finite amount of time for the flow of gas to form, which means it takes a finite amount of time for the corresponding force to be applied to the vehicle. When the valve is closed, thrust falls to zero over a finite interval. Figure 3-1 shows these characteristics. t_a is the time at which the valve is commanded on, and t_b is the time at which the valve is commanded off. t_{on} is the time at which the valve thrust reaches its max value, and t_{off} is the time at which the valve thrust falls to zero after it is commanded off.

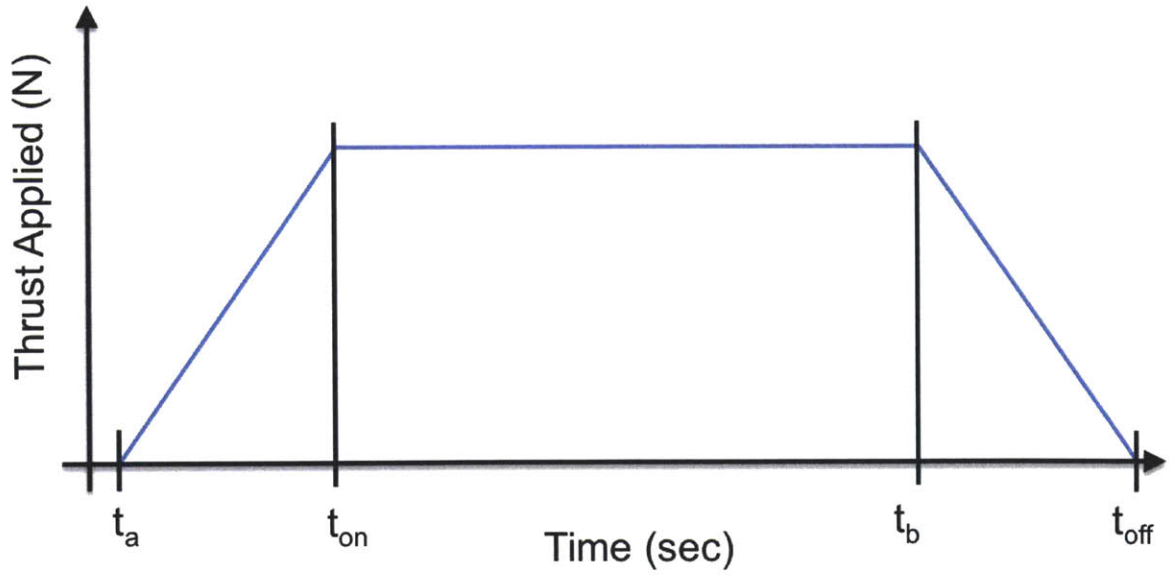


Figure 3-1: Valve Transients

This behavior can be referred to as the valve transients. In addition to significant ground testing, one key to minimizing the impact of this characteristic of the system would be to select valves for the MAV RCS thrusters which have significant space heritage. The combination of ground

testing and data from previous space missions would allow the valve transients to be effectively characterized before the mission.

The primary impact of the valve transients would be in the control law and in the stuck valve detection algorithm. Both of these subsystems must accurately predict the jet response and thrust which will be applied to the vehicle. Ineffective prediction or compensation for valve transients would result in sub-optimal control and in false or incorrect stuck valve identification. The worst case scenario would be to trigger a false valve failure indication. Therefore the algorithm must take this into account. The first step is that no failure indication shall trigger faster than it takes a valve to fully open or close. Other considerations could be incorporated into the algorithm depending on the specific nature of the valve transients. Also, the valve transients must be incorporated into the predicted change in body angular and linear rate, to be detailed below, which is a crucial step in the detection algorithm. As long as the valve transients are properly characterized, the predicted change in body rates can incorporate them and thereby prevent incorrect or missed failure indications. Because valve transient characteristics can only be determined for the particular valve and system to be used, and can be incorporated into the existing subsystems, they will not be considered in this research. Valves will be assumed to open instantaneously, with a corresponding instantaneous application of force and torque to the vehicle.

The force and torque which each valve applies upon opening, however, will not be a constant value. The force applied will vary primarily based upon the gas pressure which exists at each valve, as it is the pressure which expels the gases from the valve and therefore exerts the force on the vehicle. In most simple models, it is assumed that the pressure is constant across the entire system and that the pressure at each valve is identical. In reality this will not be the case.

The pressure at each valve may vary based on how many valves are open, and where those valves are relative to each other. Firing a neighboring valve may cause the pressure at valve j to drop significantly, however the instantaneous pressure at valve k may not be affected measurably.

Solid rocket motors can also experience instabilities resulting from acoustic and pressure oscillations.[17] This behavior has been observed in numerous space vehicles, to include the US Space Shuttle solid rocket boosters and the European Ariane 5. These oscillations have been successfully managed in these operational systems, however they do pose significant challenges to the spacecraft designers and can limit options for payload or crew capability. Significant oscillations could be dangerous to sensitive equipment and must be accurately characterized to maintain adequate vehicle control. In a gas generator RCS, this would likely manifest as uneven burning in the gas generator. For the purposes of this research, and based on [1] and [18], all MAV systems, and propulsion systems in particular will be designed with the maximum possible space heritage. This will enable the solid rocket characteristics and performance to be accurately known beforehand and therefore well compensated for in the vehicle controller and FDI and mitigation algorithms.

These fluid dynamics are too complex to fully predict, and are outside the scope of this thesis. Also, the pressure will be controlled in any such real system, and possibly also maintained at certain desired set point values for defined modes of operation. A more advanced vehicle may also have a suite of sensors to measure pressure throughout the system, possibly even at each valve. It can be assumed that the pressure set point for a given mode and any measurements which feed the pressure controller will be available to the vehicle autopilot.

Therefore this report assumes that the system and valve pressures will either be well known or well controlled and stable enough to not negatively affect the detection algorithm.

3.3 Stuck Valve Detection Algorithm

The algorithm which was chosen and developed for this research is an adapted and simplified version of Deyst and Deckert's algorithm. Algorithmic simplicity was the key criteria for selecting and developing the stuck valve detection algorithm. The method detailed in [14] is among the simplest algorithms computationally, while also being very flexible in adapting to new vehicles and component configurations, and could also be transferred to a wide range of other vehicles and applications. Keeping the computational load low allows the onboard computer to be as simple as possible, which in turn enables it to be as resistant to radiation, temperature and other disturbances as possible. Another benefit is that it requires no extra hardware, as mentioned above, providing significant savings in cost, mass and complexity.

3.3.1 Algorithm Assumptions and Variables

Some assumptions are made in adapting the algorithm for use with the hypothetical MAV for this research. As stated previously, the onboard IMU is assumed to contain all necessary filters and algorithms for producing the optimal estimate of body positions, as well as angular and linear rates and accelerations. Therefore it is not necessary to include a separate filter in the disturbance estimation algorithm as [14] did. Also, all necessary position and attitude data is assumed to be already available. Depending on the vehicle and the particular IMU, additional differencing or estimation methods may be required to produce the necessary data for disturbance estimation, however that is outside the scope of this thesis. Therefore this work also does not include the estimate error covariance in the calculations, because all estimates are conducted external to the failure detection algorithm.

As stated in Chapter 1, the two vehicle failure modes to be considered are stuck on and stuck off. Stuck on represents that the valve is firing with 100% thrust at all times, regardless of the thruster command. Stuck off means the jet is not firing, and provides 0% thrust at all times, even when the valve is commanded on.

The following variables will be used to describe the stuck valve detection algorithm. Activity vectors represent the expected instantaneous acceleration from each jet. The activity vector for each jet is a six-element vector, with the first three elements representing rotational acceleration, and the last three elements representing linear acceleration. Substituting for the variables to be used in this research, [14] begins with the failure likelihood for jet j defined as:

$$f_j = \frac{\alpha_j^T \cdot P_\beta^{-1} \cdot \beta}{\alpha_j^T \cdot P_\beta^{-1} \cdot \alpha_j} \tag{3-1}$$

where β is the estimated disturbance acceleration and P_β^{-1} is the disturbance acceleration estimate error covariance. Incorporating estimation error was necessary in the reference because vehicle angular and translational acceleration values were not available in real time and had to be estimated. This algorithm assumes that all necessary data is available in real time and does no estimation external to the IMU. Therefore the estimation error covariance P_β can be assumed to be the six by six identity vector. The likelihood of failure for jet j can then be simplified as:

$$f_j = \frac{\alpha_j^T \cdot \beta}{\alpha_j^T \cdot \alpha_j} \tag{3-2}$$

For the algorithm to be described below, two separate on and off failure likelihoods shall be maintained. f_j^{off} shall be the likelihood that jet j is stuck off and f_j^{on} shall be the likelihood that jet j is stuck on. These variables are used to set the failure indication variables, J^{fail} and P^{fail}. J^{fail} is a Boolean vector of sixteen elements, representing whether each jet has failed.

P^{fail} is also a sixteen element vector whose elements vary from 0 to 1, representing the percentage of thrust that the jet is firing with at all times, between 0% and 100%.

In order to prevent an error at a single time step from triggering a false failure indication, the variables L_1^{off}, L_2^{off}, L_1^{on}, and L_2^{on} shall represent the stuck on and stuck off pushdown lists which are used to estimate that a jet has failed. If n is the sequential number of the current time step, pushdown list L_1^{off} represents the 16-element vector of off failure likelihood estimates from time step $n-1$. The corresponding elements of each pushdown list are updated only when each individual jet is tested for either an off or an on failure, i.e. either when that jet is or is not commanded to fire, respectively. In this way, an accurate estimate of failure likelihood can be maintained during extended periods when a particular jet is not tested for one or the other type of failure.

3.3.2 Algorithm Description

The primary steps of the algorithm are as follows:

1. Set new pushdowns as old values from memory to ensure there is not an empty value in memory in case of an algorithm error or premature loop break. That is:

$$L_1^{off}(n) = L_1^{off}(n-1) \tag{3-3}$$

$$L_2^{off}(n) = L_2^{off}(n-1) \tag{3-4}$$

$$L_1^{on}(n) = L_1^{on}(n-1) \tag{3-5}$$

$$L_2^{on}(n) = L_2^{on}(n-1) \tag{3-6}$$

2. Develop an estimate of the disturbance acceleration acting on the vehicle, β. The primary method for doing this is to compare the expected change in rotational and linear body rates, $\Delta\omega^{pred}$ and Δv^{pred} respectively, with the actual change in body rates

measured by the IMU, $\Delta\omega$ and Δv. $\Delta\omega^{pred}$ and Δv^{pred} are calculated by the autopilot and mitigation logic according to equations (3-7), (3-8) and (3-9) and then given to the failure detection algorithm. C is assumed to be the consecutive list of all jets which are to be fired at the given time step, and T_S is the system sample or cycle time.

$$\beta = \begin{bmatrix} \Delta\omega - \Delta\omega^{pred} \\ (\Delta v - \Delta v^{pred}) \end{bmatrix} \tag{3-7}$$

$$\Delta\omega^{pred} = \sum_{j=C} \alpha_j(1:3) \tag{3-8}$$

$$\Delta v^{pred} = \sum_{j=C} \alpha_j(4:6) \tag{3-9}$$

3. For all jets 1 through 16

   a. If jet j is commanded, update f_j^{off}, and $L_{1,j}^{off}$ and $L_{2,j}^{off}$ for that jet

$$f_j^{off} = \frac{\left(f_j + L_{1,j}^{off} + L_{2,j}^{off}\right)}{3} \tag{3-10}$$

$$L_{2,j}^{off}(n) = L_{1,j}^{off}(n) \tag{3-11}$$

$$L_{1,j}^{off}(n) = f_j \tag{3-12}$$

   b. If jet j is not commanded, update f_j^{on}, and $L_{1,j}^{on}$ and $L_{2,j}^{on}$ for that jet

$$f_j^{on} = \frac{\left(f_j + L_{1,j}^{on} + L_{2,j}^{on}\right)}{3} \tag{3-13}$$

$$L_{2,j}^{on}(n) = L_{1,j}^{on}(n) \tag{3-14}$$

$$L_{1,j}^{on}(n) = f_j \tag{3-15}$$

   c. Set J_j^{fail}, the Boolean failure flag and P_j^{fail}, the percentage failure

      i. If $f_j^{on} > f_T^{on}$

         1. Then $J_j^{fail} = 1, P_j^{fail} = 1$

ii. If $f_j^{off} < f_T^{off}$

1. Then $J_j^{fail} = 1, P_j^{fail} = 0$

Some important notes on the stuck valve detection algorithm are as follows. First, the pushdown lists as described here are used for simplicity. Iterating the failure detection and compensating for noise or error sources could also be done with a Kalman-type filter or estimation methods such as recursive least squares At present it is assumed that the onboard IMU provides navigation data of sufficient quality that such noise considerations are not required, however this could be adapted or expanded upon in future work. Setting the threshold values is also very important. For this research, $f_T^{off} = -0.8$ and $f_T^{on} = 0.8$, so for both on and off failures 80% was considered to be the threshold for detection of a full on or off failure. This was determined to be acceptable based on the noise considerations that were made in the final simulation and the vehicle characteristics. Also, with the pushdown lists, disturbance accelerations at two successive time steps would result in failure indications f_j^{off} or $f_j^{on} = 0.667$, so the detection threshold needed to be above this value to allow the algorithm to work properly. If more pushdown lists or a noise filter were added, or if the algorithm were adapted to a new vehicle, new threshold values would need to be established, however 80% is suggested as an initial value for analysis.

Also, the disturbance vector may need to be scaled. If the translational and angular components of the individual jet activity vectors are of different scale, one may overshadow the other and dominate the likelihood estimation. A simple numerical example can illustrate this. For the hypothetical MAV, the magnitude of a jet 1's expected rotational acceleration along the Y-axis may be 13 deg/s^2, and the expected linear acceleration may be 0.2 m/s^2 along the X-axis, nearly two orders of magnitude smaller in simple numerical value. Changing the units, such as

using radians instead of degrees, could also have a similar effect. In an on failure situation, the vehicle may be experiencing disturbance acceleration which matches the failure signature of jet 1, and produces a failure indication f_1^{on} of 1.000 for that time step. Because the rotational component of jet 7's activity vector is nearly identical to that of jet 1, the failure indication for jet 7 would equal 0.9997 in this case. This is essentially indistinguishable from the correct failure indication for valve 1, and is obviously above the threshold for detection, so therefore valve 7 would be flagged as failed along with valve 1. This shows that the choice of units and relative magnitudes for rotation and translation could produce false or inaccurate failure indications.

Therefore the translation and rotation elements of both the disturbance and activity vectors must be scaled by their relative magnitudes in order to balance the failure likelihood calculation and adequately distinguish each jet. In all cases, the change to the disturbance vector can be calculated as follows:

$$\varphi_j = \frac{|\alpha_j(1:3)|}{|\alpha_j(4:6)|} \tag{3-16}$$

$$\beta' = \begin{bmatrix} \Delta\omega - \Delta\omega^{pred} \\ (\Delta v - \Delta v^{pred}) \cdot \varphi_j \end{bmatrix} \tag{3-17}$$

$$\alpha_j' = \begin{bmatrix} \alpha_j(1:3) \\ \alpha_j(4:6) \cdot \varphi_j \end{bmatrix} \tag{3-18}$$

where $\Delta\omega^{pred}$ and Δv^{pred} represent the predicted change in body angular and linear rate, respectively, and are calculated based on which jets are selected to be fired by the autopilot. β' is the augmented disturbance vector and α_j' is the augmented jet activity vector. The augmentation ratio φ_j is calculated separately for each jet because the disparity between linear and angular control authority is different for each jet. In some vehicles, such as the hypothetical MAV in this research, jets of the same type, such as all those that produce X-axis linear

acceleration and Y and Z-axis angular acceleration, may have the same augmentation ratio. If vehicle mass properties are not constant, particularly the inertia tensor and center of mass distribution, the augmentation ratio may change for individual jets relative to others, and therefore the ratios must be calculated for each jet at each time step.

Based on the above equations, Equation (3) can be updated as follows:

$$f_j' = \frac{\alpha_j'^T \cdot \beta}{\alpha_j'^T \cdot \alpha_j'} \tag{3-19}$$

Where f_j' represents the augmented failure indication. Equations (3-10), (3-12), (3-13) and (3-15) can then be augmented to include f_j' in place of f_j.

3.4 Valve Isolation

One key problem faced by many FDI systems is not only determining that a fault or failure has occurred, but also isolating that failure to its correct source. This is the purpose of the separate off and on failure detection variables in Section 3.3, in order to isolate which jet has failed. During one particular time step, a failed on jet will produce the exact same disturbance acceleration as if its opposing jet is commanded on but is failed off. Since firing opposing thrusters simultaneously produces no net force or torque and only expends unnecessary fuel, it can be assumed that any viable autopilot and jet selection method would not command this directly. Opposing thrusters may be fired to release system pressure without applying any net force or torque to the vehicle, and this particular behavior could be added to the predicted body motion calculations and other elements of the algorithm. With this assumption, testing jets for on failures only when they are not commanded, and testing jets for off failures only when they are commanded, enables the direct isolation of which jet has failed and what failure mode it is experiencing, either on or off.

3.5 Multiple Simultaneous Failures

The arrangement of valves for the simulation is described in Chapter 5. The first eight valves control motion in the X-axis of translation and the Y and Z-axes of rotation and may be referred to as type 1 valves. The second eight valves control motion in the Y and Z-axes of translation and the X-axis of rotation, and may be referred to as type 2 valves. As previously stated, this research is limited to detecting only a single failure at any one time. The current algorithm is, however, capable of detecting and correctly identifying certain combinations of simultaneous failures, because the activity vectors of type 1 and 2 jets are completely disparate in six-dimensional space. An on failure of a type 1 jet can be detected at the same time as an off failure of a type 2 jet, and an off failure of a type 1 jet can be detected at the same time as an on failure of a type 2 jet.

It is obvious from the previous discussion that simultaneous on and off failures of jets of the same type cannot be accurately detected, because at a given time step an off failure of a jet appears identical to an on failure of its opposing jet. The ability to detect these particular types of simultaneous on and off failures is an artifact of the current algorithm design and is not sufficient to say that the algorithm can adequately handle multiple simultaneous failures. It adds a small level of robustness to the algorithm and shows the potential for extension of these algorithms to the detection and mitigation of multiple simultaneous failures.

4.0 Failure Mitigation

4.1 Disturbance Mitigation

Disturbance mitigation should be handled differently for the on and off failure modes. The primary issues to consider with each failure mode are: how to counteract (mitigate) the disturbance acceleration from the failure, how best to remove the failed jet from the simplex selection options, and how to deal with intermittent failures.

4.1.1 Stuck On Failure

As described before, the hypothetical MAV has sixteen jets arranged in eight opposed pairs. For on failures, the most obvious solution is to fire the opposing jet continuously in order to oppose the disturbance. For jet selection, in this case both the failed and the opposing jet would be removed from the jet options in Simplex. This does, however, present a problem, because removing two jets places extra constraints on Simplex, and in some cases Simplex may be unable to find a solution for the desired rate change request. Also, the vehicle autopilot may wish to take the vehicle in the direction which the on failed jet is thrusting, however the opposing jet's mitigation is removing any possible benefit.

The solution to this is to alter the mitigation and jet selection slightly to take maximum advantage of the stuck jet and to limit the constraints on Simplex. In this new scheme, the opposing jet is still fired, and only that opposing jet is removed from the options in Simplex. The failed jet itself remains as an option for Simplex to choose. If the failed jet is commanded, the opposing jet is turned off for as long as the failed jet is commanded on. Therefore the net force and torque on the vehicle at that moment are the same as commanding the failed jet, and

the opposing jet fires only when necessary to counteract unwanted disturbance. In this way, the vehicle can achieve the maximum possible controllability in the presence of a failure.

The problem of intermittent on failures is significant. Continuously firing the opposing jet when the failed jet disturbance has gone away would be as detrimental to vehicle performance and controllability as was the original failure. Therefore the detection and mitigation algorithms as previously described are capable of detecting and correcting for intermittent on failures, and an example of this is included in the simulation section. If the on failure stops and the jet goes back to normal, the disturbance acceleration goes away. The opposing jet does not appear as a disturbance because the autopilot incorporates the expected acceleration from the opposing jet into $\Delta\omega^{pred}$ and Δv^{pred} in order to maintain the correct disturbance estimation even when the opposing jet is exactly counteracting the on failed jet. As the disturbance from the failed jet goes away, so does the failure indication f_j^{on}. Therefore the opposing jet is turned off as soon as the failure indication f_j^{on} drops below the threshold f_T^{on-}, a new threshold value which is lower than f_T^{on}. If the opposing jet were turned off every time the failure indication f_j^{on} dropped below the original threshold f_T^{on}, an on failure with thrust of the same percentage as the threshold value would cause the failure flag to oscillate, producing erratic and undesirable vehicle behavior. Following this the vehicle returns to normal operation until another failure is detected.

4.1.2 Stuck Off Failure

The primary method for stuck off failure mitigation is simply to remove that jet from the list of options available to the Simplex jet selection algorithm. The effect of a stuck off failure is less dramatic than a stuck on failure because the disturbance only appears when the jet is commanded. When an off failure is detected, however, it is important that the jet is taken away from Simplex in order to improve performance. If the jet is not removed from the list of jet

options in Simplex, Simplex will likely try to fire that jet, since it acts directly along the disturbance vector which its failure has created. This can have very detrimental consequences for vehicle control, as evident in Section 5.3.5.

The algorithm as currently implemented and described here is not capable of detecting that a previously failed off jet has returned to normal operation. A simple update to the mitigation strategy would allow for intermittent off failure correction. A test can be implemented to determine if Simplex cannot find a solution to the current rate change request with the available jets. If this test is triggered, the off failure flag for that jet can be reset in order to give the jet back to Simplex. If the jet is subsequently commanded and does not fire, the disturbance acceleration will appear again and the failure flag will again be reset. If the jet is commanded and does fire, the vehicle can continue and return to normal operation. The off failed jet may or may not be the reason that Simplex is unable to find a solution for the current rate change. Returning the failed jet in this way ensures that an off failure detected early in a mission does not permanently remove the jet from use if the jet returns to normal functionality, and increases the chances of Simplex finding a viable solution to a given rate change request. An alternative option would be to design dedicated sets of jet firings which could be triggered as necessary by the autopilot in order to detect off failed jets and determine if a previously off failed jet has been restored.

5.0 Simulation

5.1 Simulation Description

Research simulations were conducted using Matlab and Simulink. The heart of the simulation is a generic six degree-of-freedom equations of motion simulator developed at the Charles Stark Draper Laboratory. The simulator receives inputs of force F and torque or moment M acting on the body, and parameters of vehicle mass m and inertia I, and derivatives $\frac{dm}{dt}$ and $\frac{dI}{dt}$. The vehicle mass properties can be changed during a test run by either changing the values of m and I as the simulation progresses, or setting nonzero values for $\frac{dm}{dt}$ and $\frac{dI}{dt}$.

As described in Chapter 1, the proposed representative Mars Ascent Vehicle has sixteen jets arranged into eight opposed pairs, which are capable of controlling motion in any of the six degrees of freedom. The vehicle is one redundant, meaning that any one failure does not result in loss of vehicle control, however there exists at least one set of two failures that could render the vehicle uncontrollable in at least one degree of freedom.

The MAV as described in [18] will likely be a nearly-cylindrical, two-stage rocket approximately 2.5 to 3 m tall. Some versions of the MAV are proposed to be stored horizontally in the base of the lander for space conservation and sample loading and then lifted vertically for launch. The first stage rocket will boost the payload off the Mars surface and on a trajectory for Mars orbit. The second stage will be responsible for placing the sample into the proper orbit for rendezvous with the Earth Return Vehicle (ERV). The most recent published design calls for the second stage to be a solid rocket motor with limited orbit accuracy and a small liquid RCS. This research considers the second stage as a six degree-of-freedom RCS for greater orbit accuracy and control, and for greater applicability of the algorithms to other vehicle applications.

The configuration considered here shall be a narrow cylinder, with length 1.3 m and radius 0.35 m, with the sample along the long axis near the nose, the sixteen jets arranged around the edge of the cylinder, and the electronics and gas generator hardware inside the cylinder. The vehicle X axis shall be along the centerline of the body's long axis, with the positive X axis being nominally vertical at the time of MAV launch. The positive Y axis shall bisect the vehicle IMU, and the Z axis shall be orthogonal to both the X and Y axes. In order to achieve orbit control and accurately place the Mars sample into orbit, eight of the jets (four of the opposed pairs) will be larger than the other eight, and will be pointed along the X axis. In this way, they shall be capable of controlling linear motion along the X axis, and rotation about the Y and Z axes. The eight smaller jets shall be placed along the edge of the vehicle, perpendicular to the X axis, in order to control Y and Z axis linear motion and X axis rotation. The following table shows the numbered jets and which axes they exert force or torque along or about. Appropriate values for mass and jet force are used, based on [18]:

Jet Number	Force	Torque	Jet Number	Force	Torque
1	+X	+Y, -Z	9	-Y, +Z	+X
2	+X	+Y, +Z	10	-Y, -Z	+X
3	+X	-Y, +Z	11	+Y, -Z	+X
4	+X	-Y, -Z	12	+Y, +Z	+X
5	-X	-Y, +Z	13	+Y, -Z	-X
6	-X	-Y, -Z	14	+Y, +Z	-X
7	-X	+Y, -Z	15	-Y, +Z	-X
8	-X	+Y, +Z	16	-Y, -Z	-X

Table 5-1: Jet Forces and Torques

5.2 Test Methodology

For the test cases detailed here, the simulation is run at a sample rate of $T_s = 100\ Hz$. The phase space control law, Simplex jet selection method, and failure detection algorithm are all run at this same rate. A separate function is used to convert firing times, jet lists, and jet failure indications into correct force and torque applications to the vehicle equations of motion simulator.

The phase space control law is implemented only to zero rate and position errors and return the vehicle to the desired state. Maneuvers are executed by changing the vehicle desired state at specific points in time. Detection performance is evaluated primarily on two criteria: detection accuracy and detection time. If a failure is correctly identified, the goal detection time is one second, in order to prevent loss of vehicle control and allow sufficient time for mitigation to take effect. The larger jets apply approximately 13 deg/s^2 in angular acceleration about the Y and Z-axes. Unmitigated, this will result in an undesired vehicle motion of 6.5 degrees after one second, and 26 degrees after two seconds. If it is assumed that the mitigation will take effect within one second after the failure flag is triggered, then a detection time of one second means that a failure should be fully mitigated within two seconds. This ensures that the vehicle is back under control before it has departed significantly from its desired course.

Appropriate amounts of noise were added to the simulated IMU measurements to represent a real system. The intent is to show that the algorithms still function well in the presence of noise and imperfect measurements. The noise is implemented as band-limited white noise, added to the linear and angular acceleration measurements within the six degree of freedom equations of motions simulator. The acceleration measurements are then integrated to

produce the linear and angular rates and positions, thereby carrying the noise into those measurements as well.

5.3 Test Cases

Unless otherwise stated, the test cases detailed here are based on the following maneuver. The maneuver begins with a +Y rotation, followed by +Z rotation, and then the failure occurs at $t = 30\ sec$. The failure is followed by +X translation, 50 second hold, then a three axis rotation with +X, -Y (return to 0 degrees), and –Z (return to 0 degrees). The final maneuver is a – X translation to return to 0 meters at $t = 150\ sec$.

5.3.1 Nominal Case

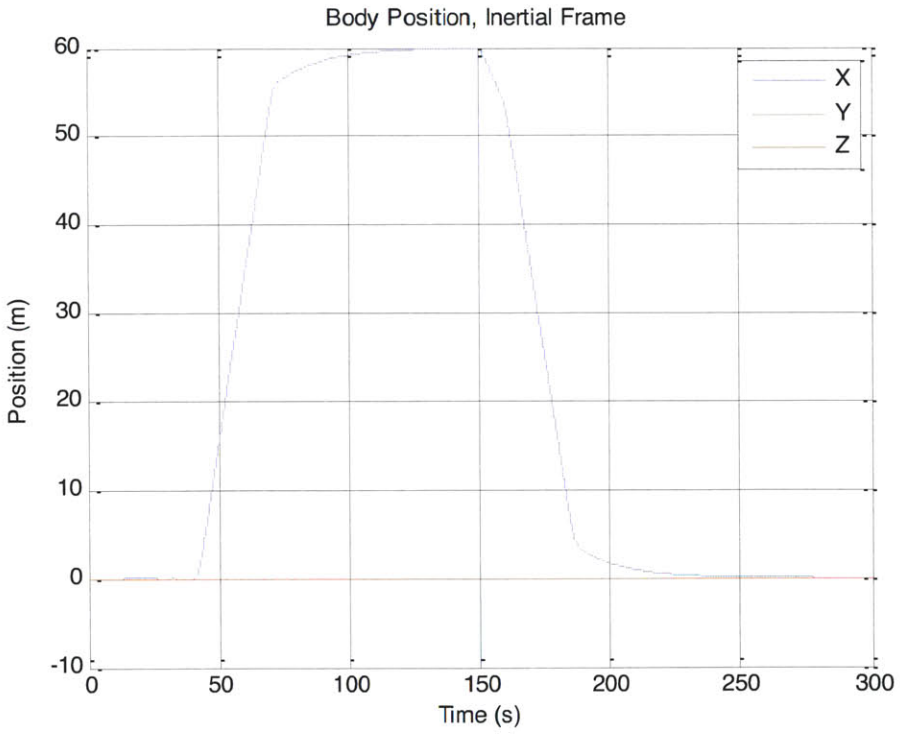


Figure 5-1: Vehicle Linear Position

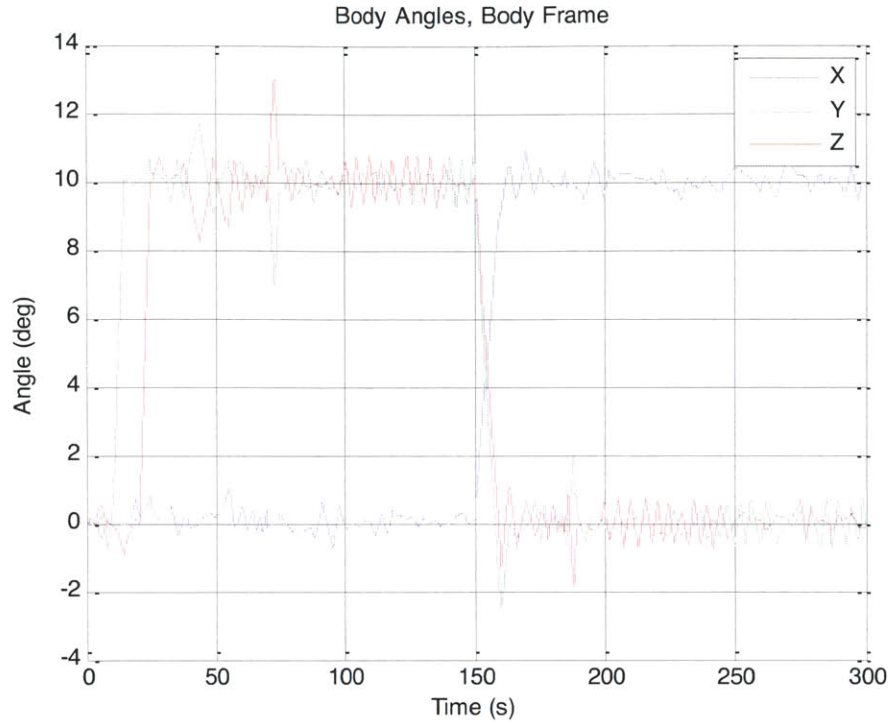


Figure 5-2: Vehicle Angular Position

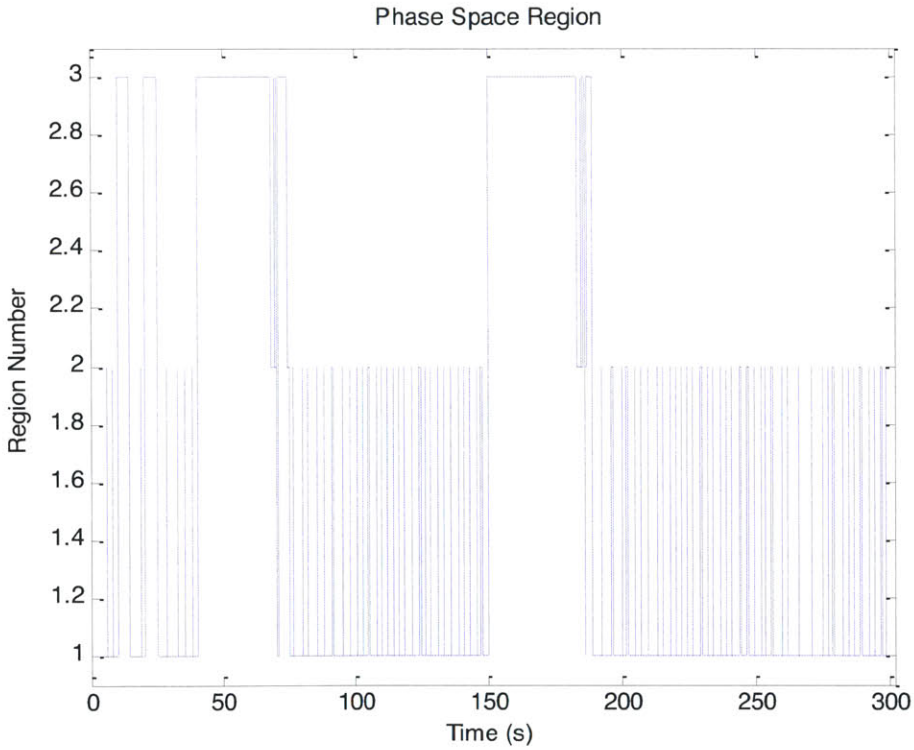


Figure 5-3: Phase Space Region

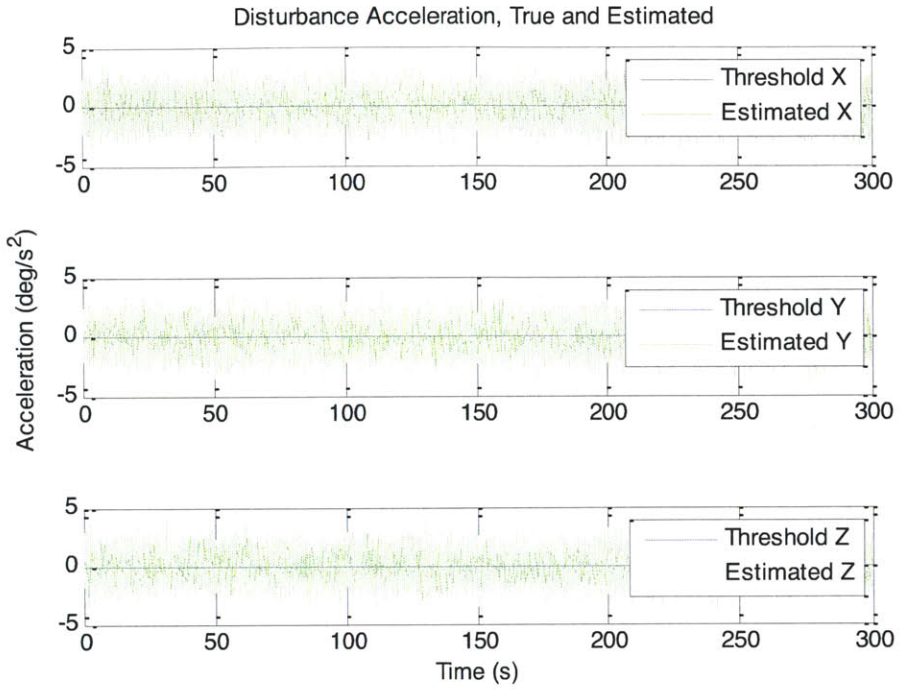


Figure 5-4: Angular Disturbance Acceleration

The nominal, no failure case is shown to establish a baseline against which to compare the remaining simulations and test cases. A few important characteristics should be noted. First, the reader should note the shape and pattern of Figures 5-1 and 5-2. The same maneuver is used for the following test cases unless otherwise noted. Also, readers who are not familiar with the phase space concept should briefly study Figure 5-3 in order to accurately interpret the meaning. Remember that region 3 is outside the deadband, and region 2 is the outer perimeter of the deadband which triggers a new jet selection request. In this no failure case, the excursions into region 3 occur when a new desired state is set, i.e. a maneuver is commanded. The oscillations between regions 1 and 2 are evidence of the phase space controller limit cycling behavior. Here the vehicle states are not precisely controlled at their desired value, but instead move out into region 2 before being controlled back towards the desired state, and then overshooting or being

disturbed away from the desired state. Finally, the noise characteristics can be seen easily in the nominal case. The measurement noise was implemented in the vehicle translational and linear accelerations, so therefore it is most evident in the acceleration plot Figure 5-4. The noise is less evident in the position plots Figures 5-1 and 5-2 because much of it has been integrated, which tends to smooth out the noise. The noise is not an insignificant consideration, however, when estimating vehicle disturbance and thruster failures, and will be discussed in the following simulations.

5.3.2 On Failure

The first test case is of valve 3 failed on:

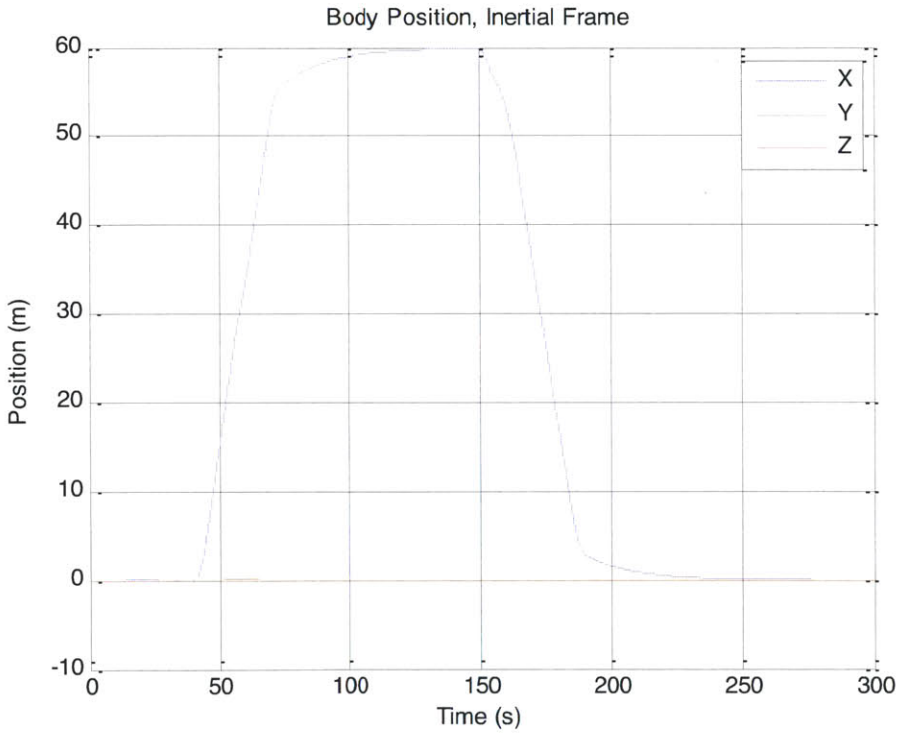


Figure 5-5: Vehicle Linear Position

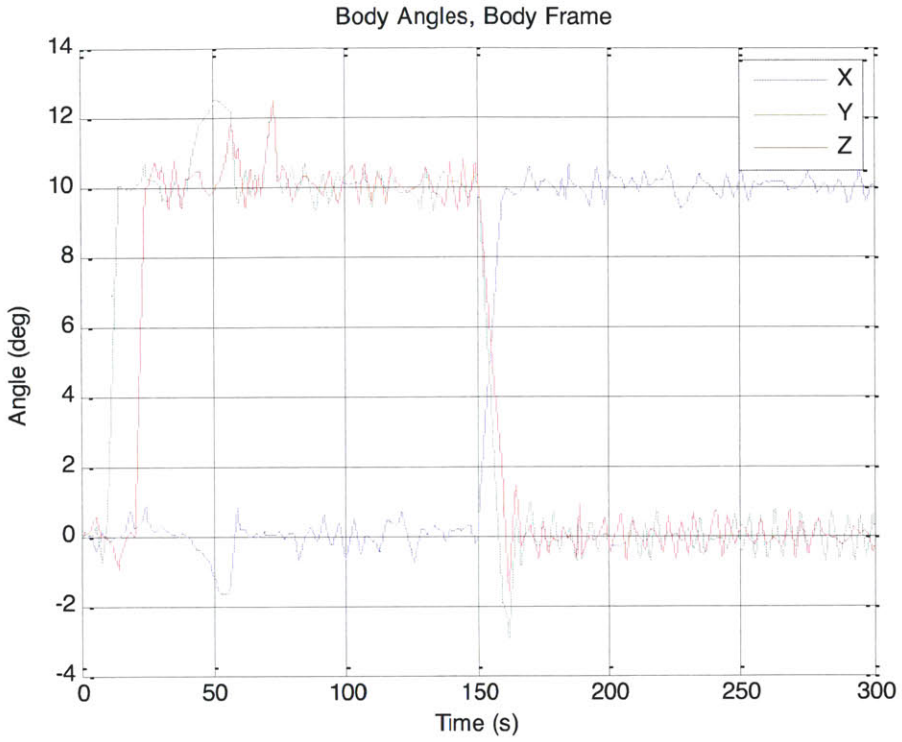


Figure 5-6: Vehicle Angular Position

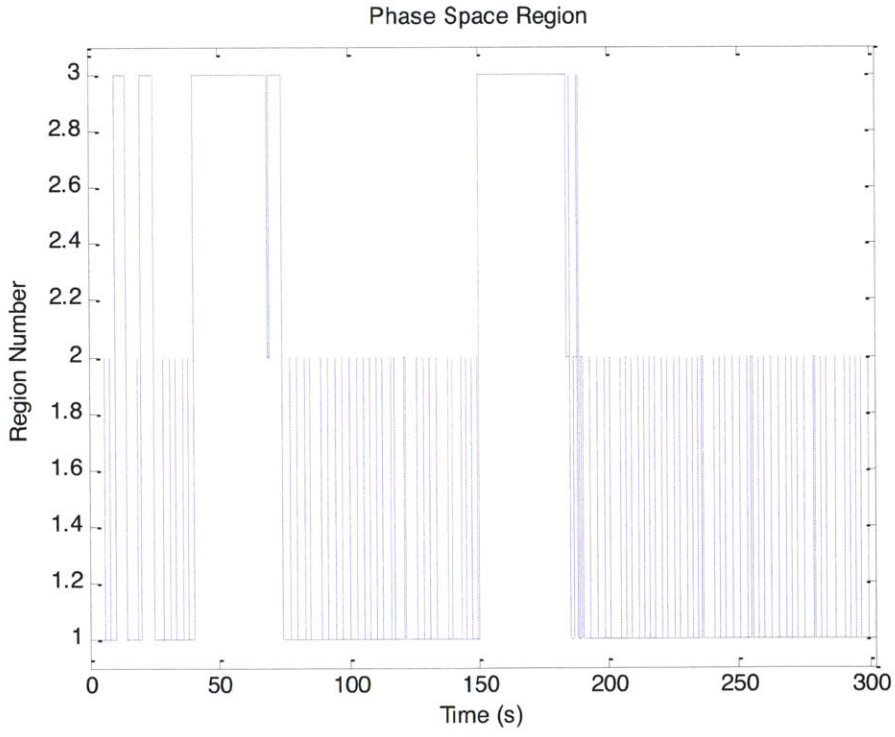


Figure 5-7: Phase Space Region

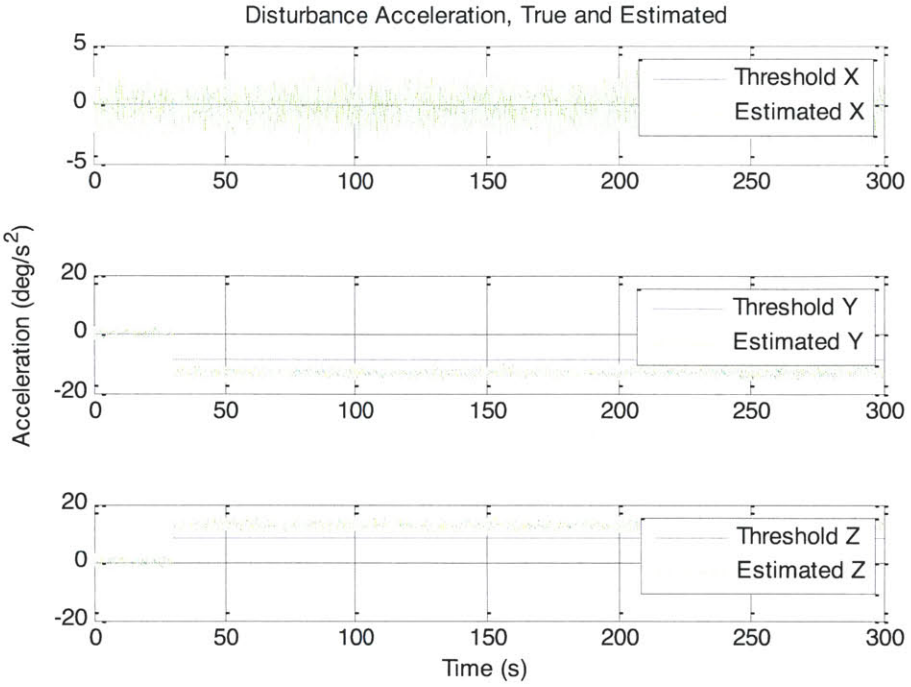


Figure 5-8: Angular Disturbance Acceleration

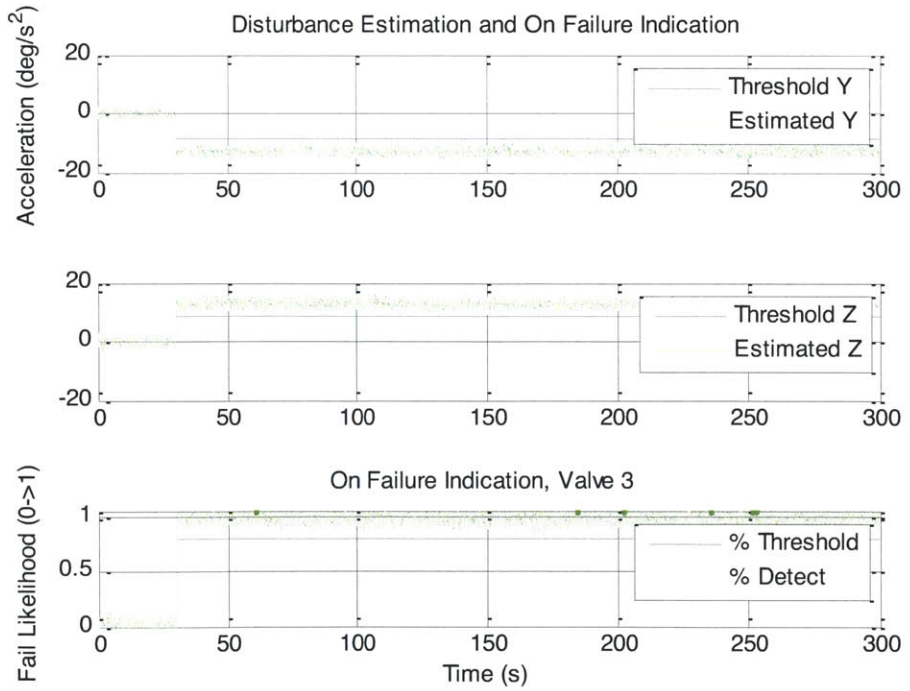


Figure 5-9: Angular Disturbance and Failure Indication

Figure 5-8 shows the rotational disturbance acceleration estimate, and Figure 5-9 shows the rotational disturbance acceleration estimates in Y and Z rotation, the axes about which valve 3 exerts torque. It is evident from these figures that the failure occurs at $t = 30\ sec$, and that the algorithm detects the failure almost immediately. The mitigation is successful, which is obvious from Figures 5-5, 5-6 and 5-7. Figure 5-5 shows the vehicle linear position in inertial space, and Figure 5-6 shows the body angular position, measured in the body frame. Figure 3 shows the vehicle's position relative to the three phase space regions. The success of the mitigation is evident from the lack of disturbance in the simulation figures. A significant disturbance is not evident in translation or rotation at $t = 30\ sec$, and the vehicle never leaves region 1 or 2 at this time, meaning that none of the six position states have exceeded their deadband.

5.3.3 Intermittent On Failure

The algorithms are also capable of detecting and properly mitigating an intermittent on failure, that is, an on failure that is only temporary and returns to normal operation after a period of time.

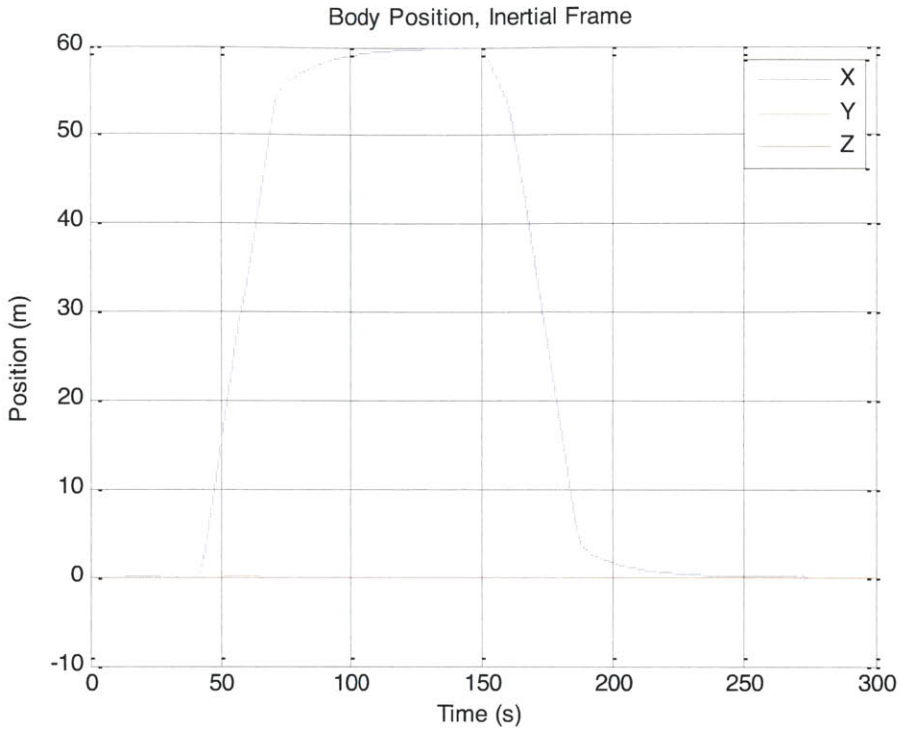


Figure 5-10: Vehicle Linear Position

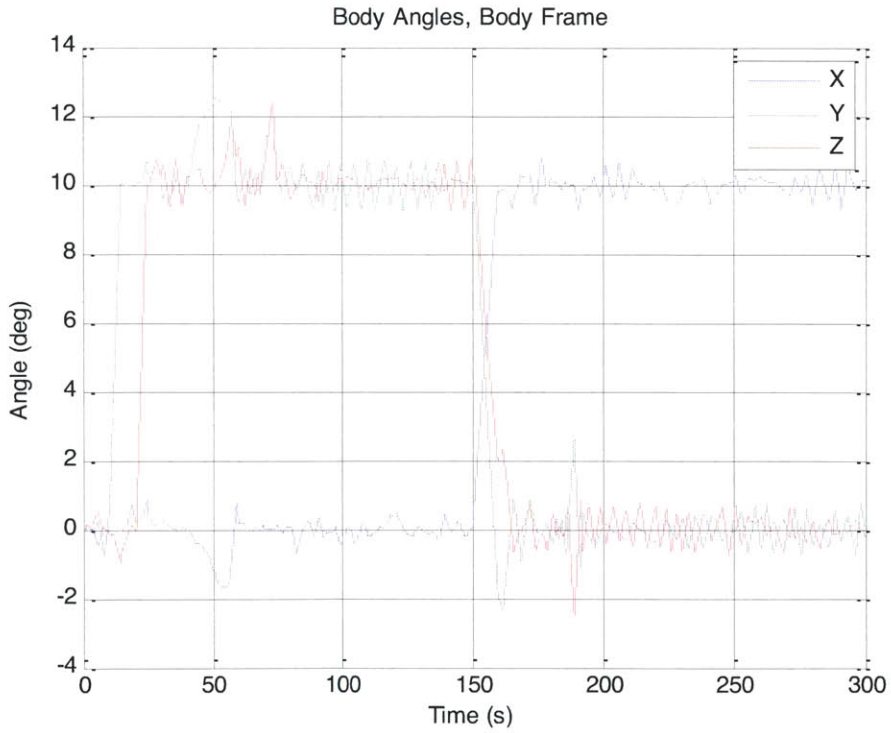


Figure 5-11: Vehicle Angular Position

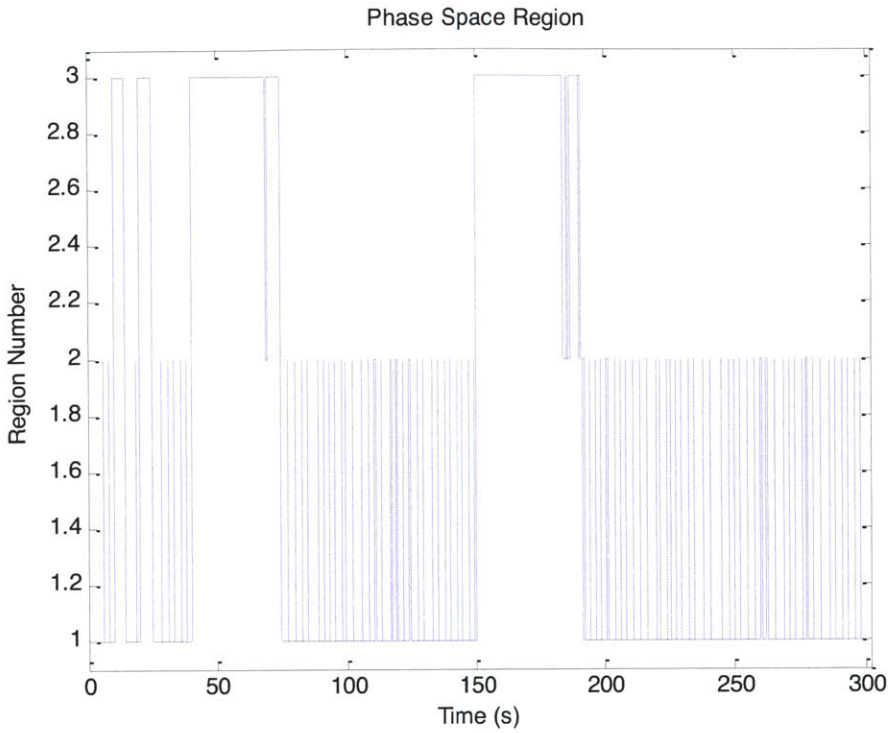


Figure 5-12: Phase Space Region

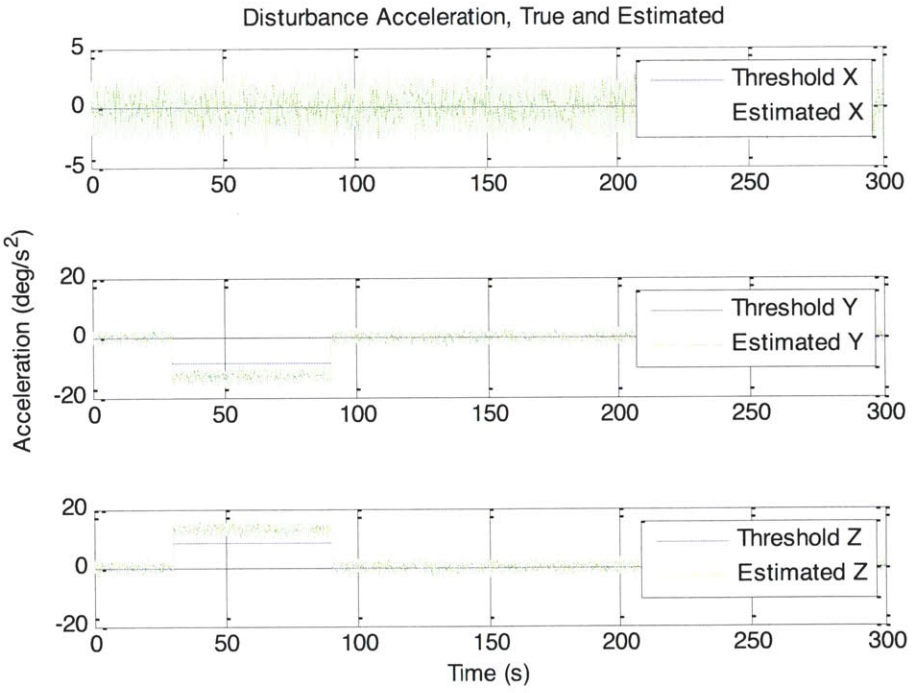


Figure 5-13: Angular Disturbance Acceleration

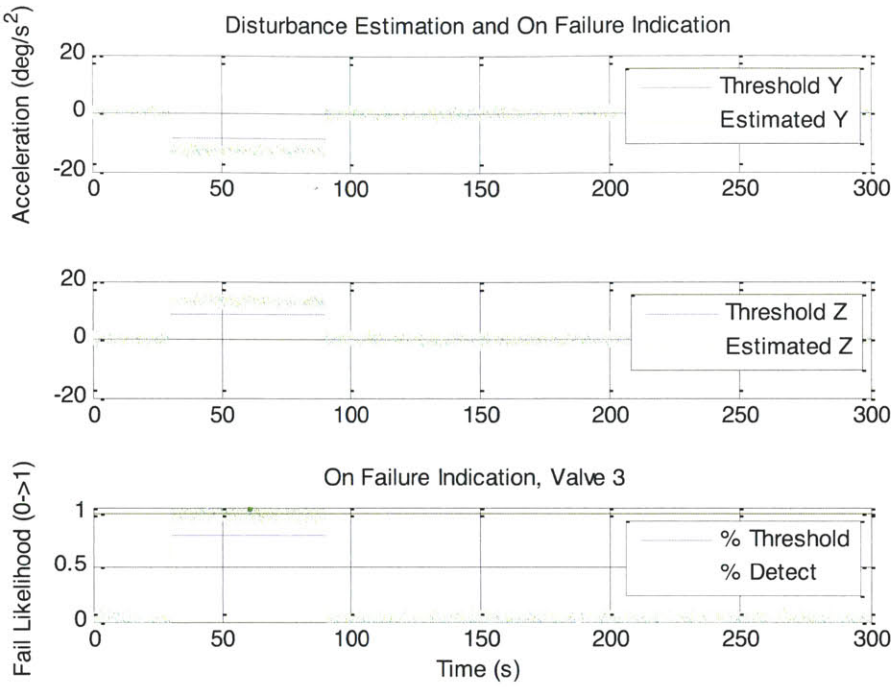


Figure 5-14: Angular Disturbance and Failure Indication

As is evident from the Figures 5-13 and 5-14, valve 3 is failed on from time $t = 30\ sec$ to $t = 90\ sec$, after which it returns to normal operation. If the detection and mitigation algorithms were not capable of identifying the change in disturbance acceleration and valve 3's return to normal operation, the opposing jet would continue to fire unnecessarily, and would itself cause a disturbance acceleration on the vehicle. Because the intermittent failure is adequately detected, the vehicle is able to return to nominal operation following the failure.

5.3.4 Valve 5 Failed Off

The next test case is valve 5 failed off while executing the same maneuver as above:

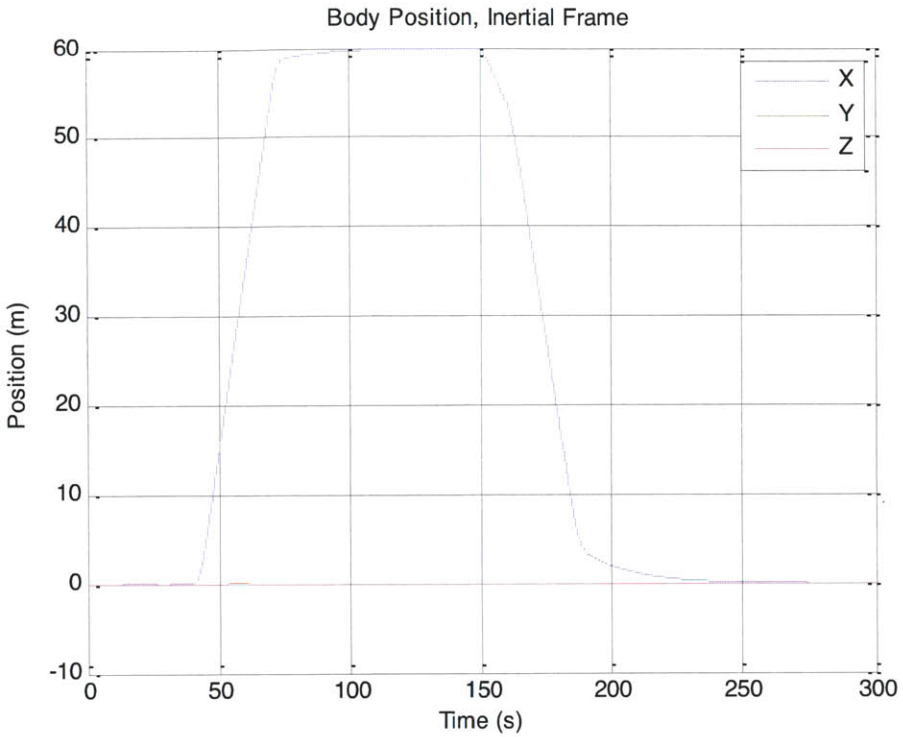


Figure 5-15: Vehicle Linear Position

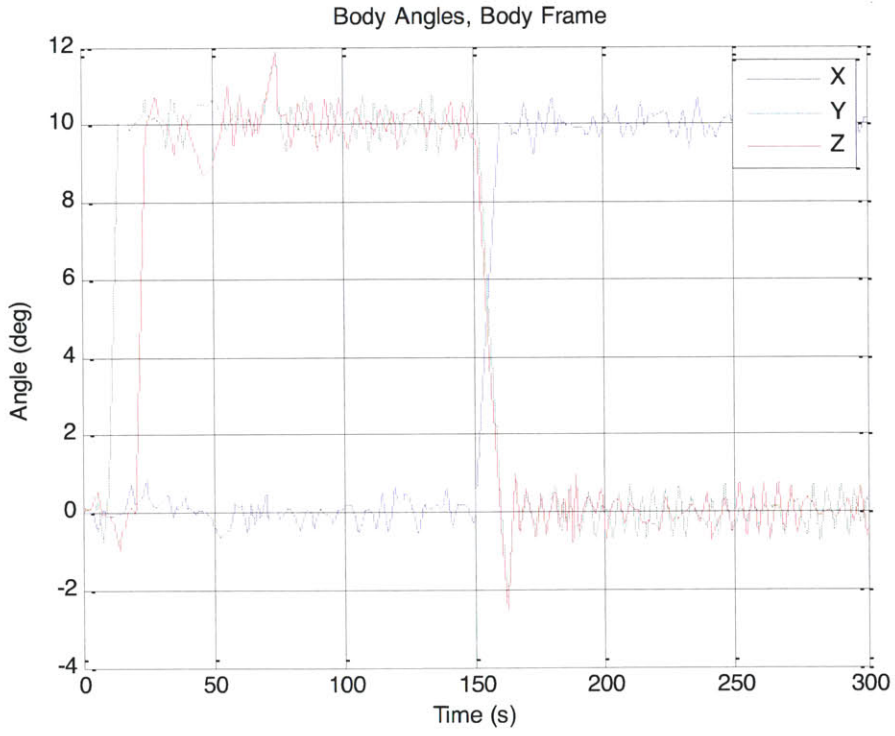


Figure 5-16: Vehicle Angular Position

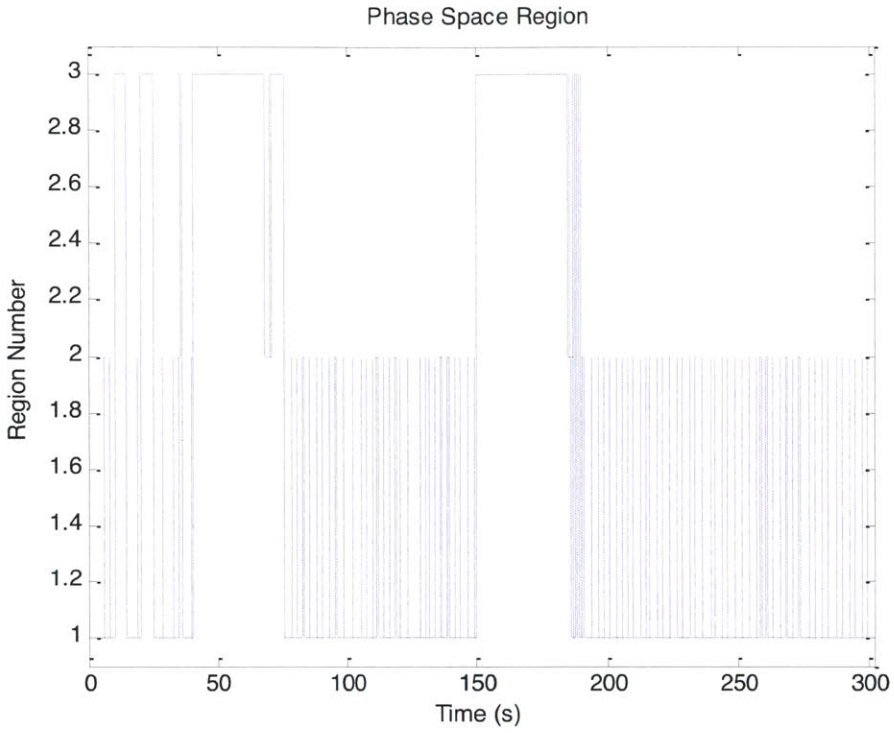


Figure 5-17: Phase Space Region

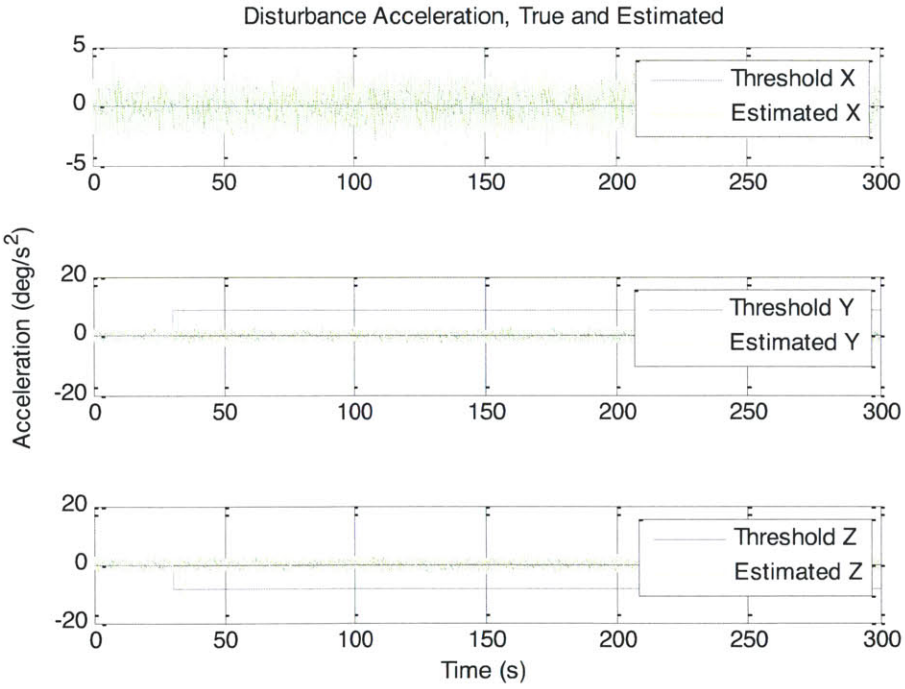


Figure 5-18: Angular Disturbance Acceleration

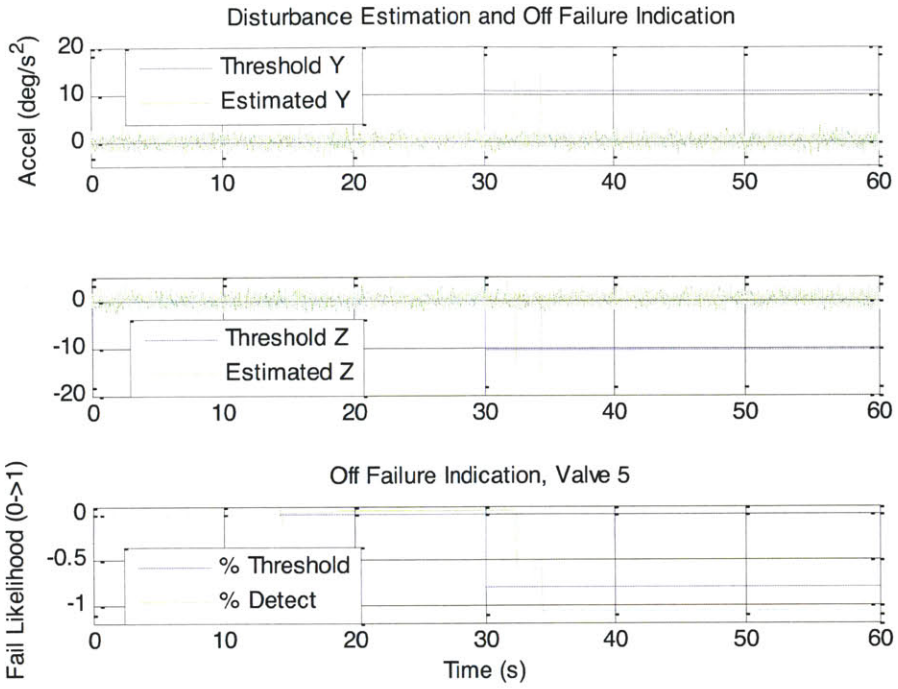


Figure 5-19: Angular Disturbance and Failure Indication

Here the failure occurs at the same time while executing the same series of maneuvers as in Section 5.3.2 above. What is most interesting to note is that the failure indication does not trigger immediately, because the off failure mode is only evident when the failed valve is commanded. The time scale of Figure 5-19 is shortened to better show this. Valve 5 is commanded on for two time steps soon after the failure, so the likelihood of failure $f_5^{on} = 0.667$, however not above the detection threshold $f_T^{on} = 0.8$, so the failure indication flag and is not set and the valve remains available for the jet selection algorithm. A few seconds later, valve 5 is again commanded on, and because the jet does not fire, the corresponding disturbance acceleration appears for the third time, the likelihood of failure for valve 5, $f_5^{on} = 1.0 > f_T^{on} = 0.8$ and the off failure indication flag is set for valve 5, that is $J_5^{fail} = 1$ and $P_5^{fail} = 0$.

Following this, valve 5 is removed from the jet selection algorithm and it is not selected again throughout the remaining time of this simulation.

5.3.5 Off Failure Not Removed from Simplex Selection,

The following example shows what happens when an off failure is not removed from the choices for the Simplex basis.

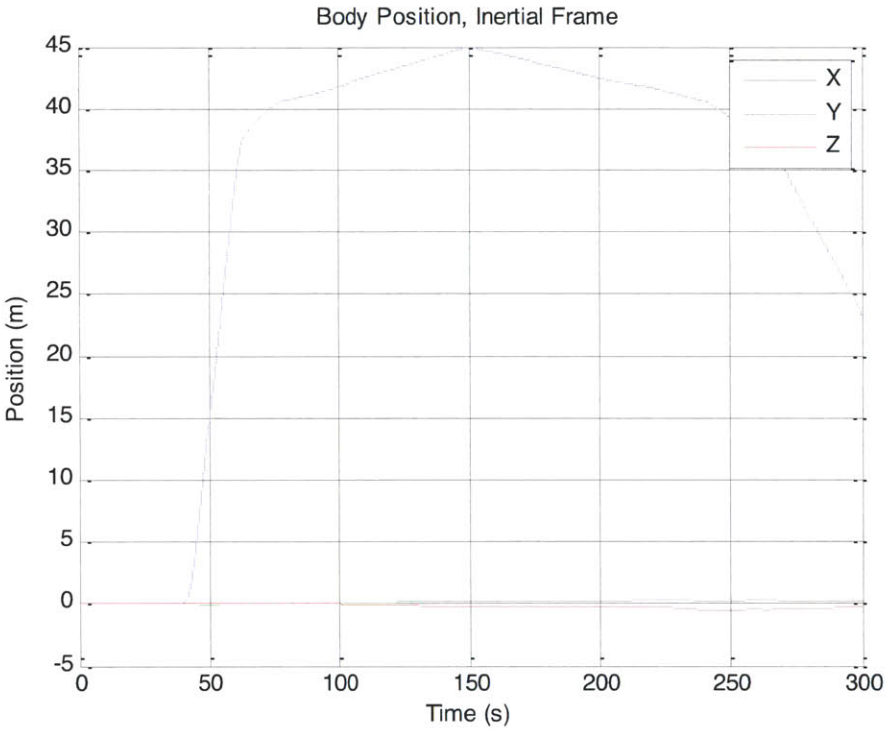


Figure 5-20: Vehicle Linear Position

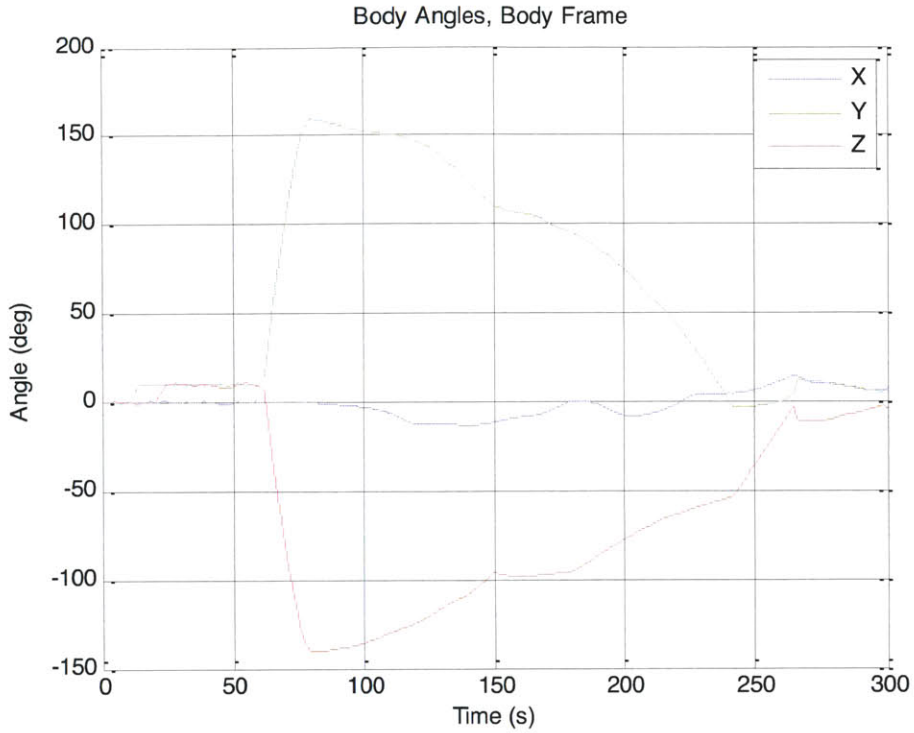


Figure 5-21: Vehicle Angular Position

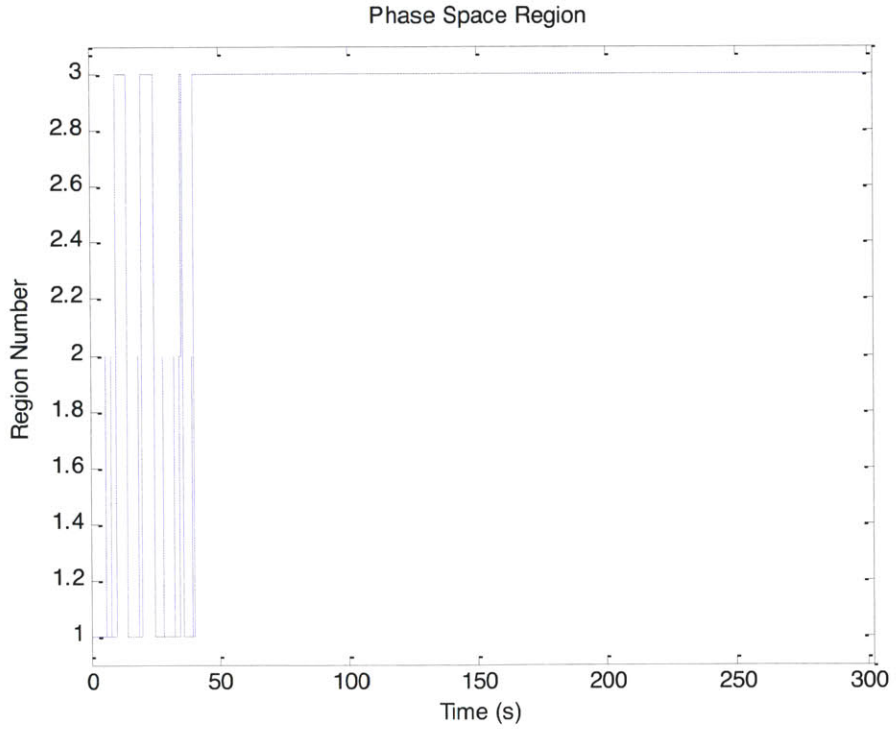


Figure 5-22: Phase Space Region

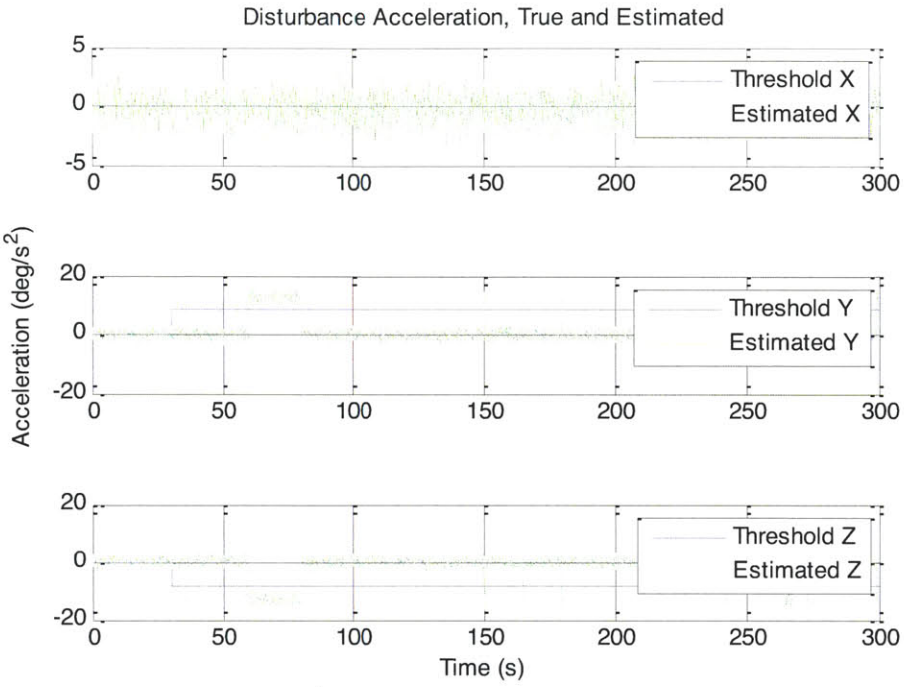


Figure 5-23: Angular Disturbance Acceleration

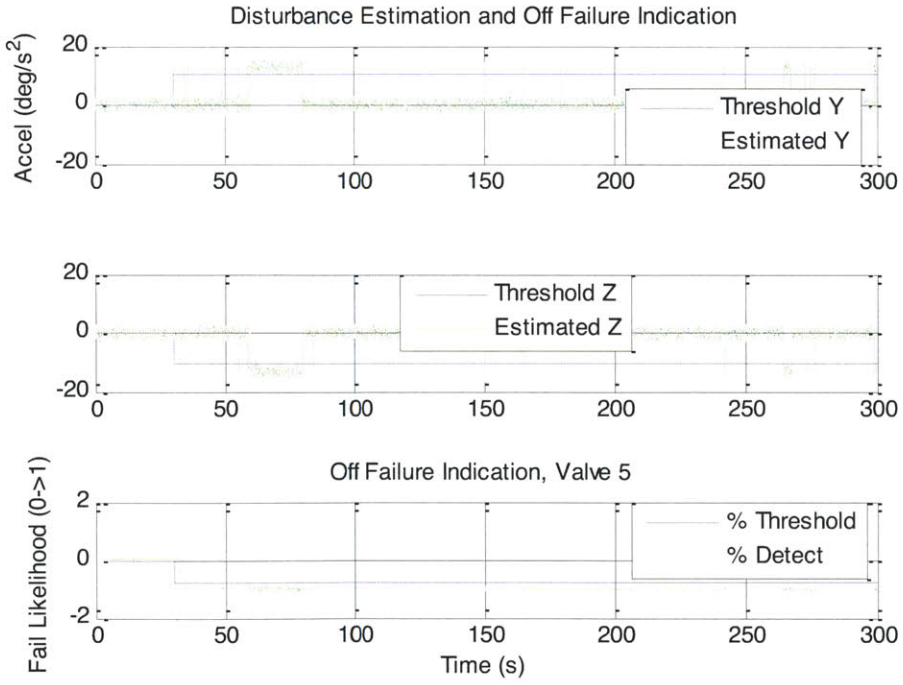


Figure 5-24: Angular Disturbance and Failure Indication

On initial inspection an off failure appears to be a relatively minor inconvenience to the system, because it is a passive disturbance that only appears whenever that valve is commanded open. The above plots show, however, that removing the failed jet from the Simplex selection basis is an essential step. If the jet is not removed, certain formulations of Simplex, such as the one used for this research, may continue to command the jet and be unable to achieve the desired maneuver. There are obviously other combinations of jets which could have accomplished the maneuver, however the failed jet was heavily favored by Simplex, and therefore the vehicle is not able to accomplish the desired maneuver.

5.3.6 Asymmetric Mass Properties

In the previous examples the mass distribution of the MAV is assumed to be fairly uniform. For the following example, the mass is distributed asymmetrically, as if the Mars sample is exceptionally large. The mass properties are also updated in the flight control and failure detection algorithms, which is essential for maintaining adequate control of the vehicle and accurately detecting valve failures. Here valve 3 fails on thirty seconds into the simulation, just as several of the previous examples.

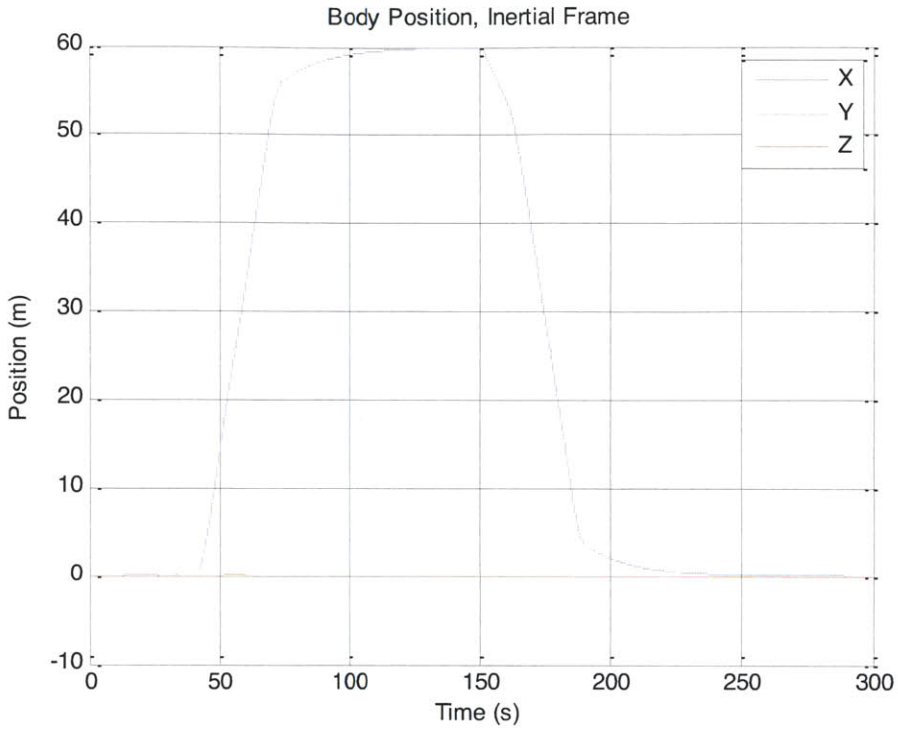


Figure 5-25: Vehicle Linear Position

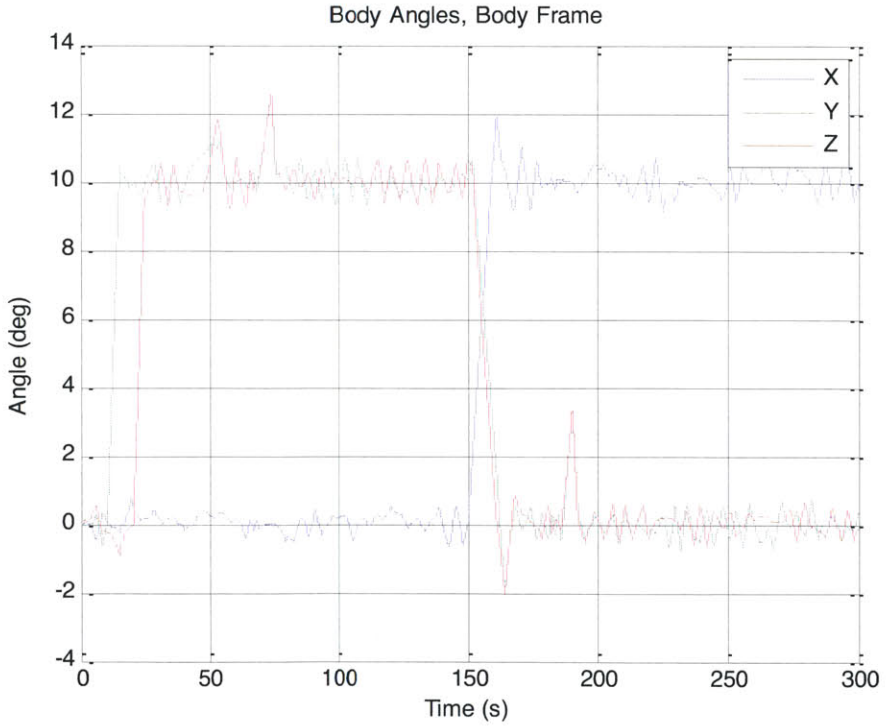


Figure 5-26: Vehicle Angular Position

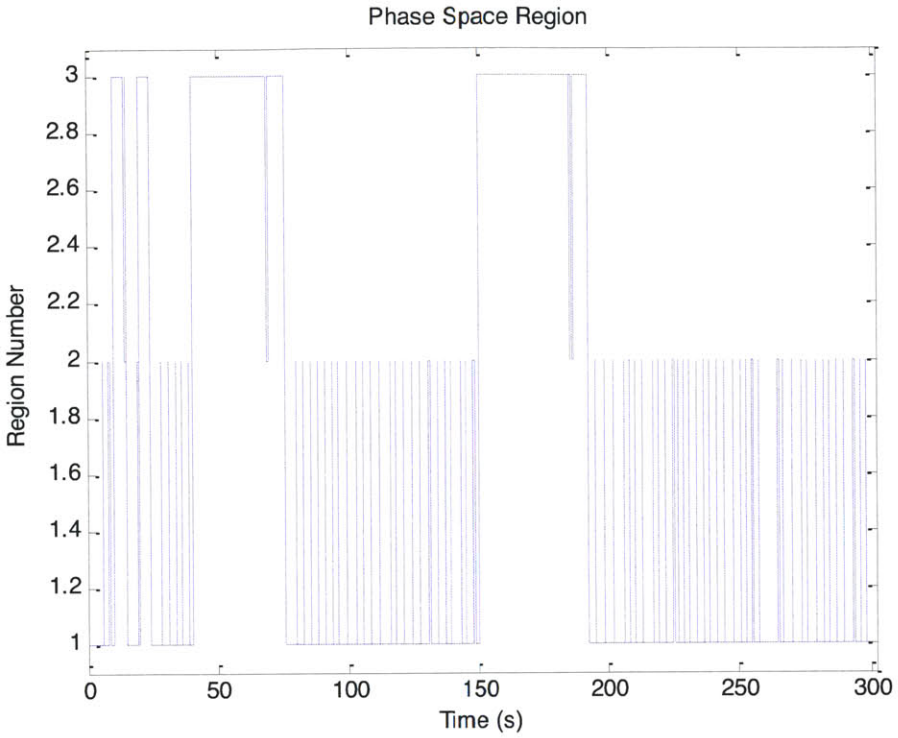


Figure 5-27: Phase Space Region

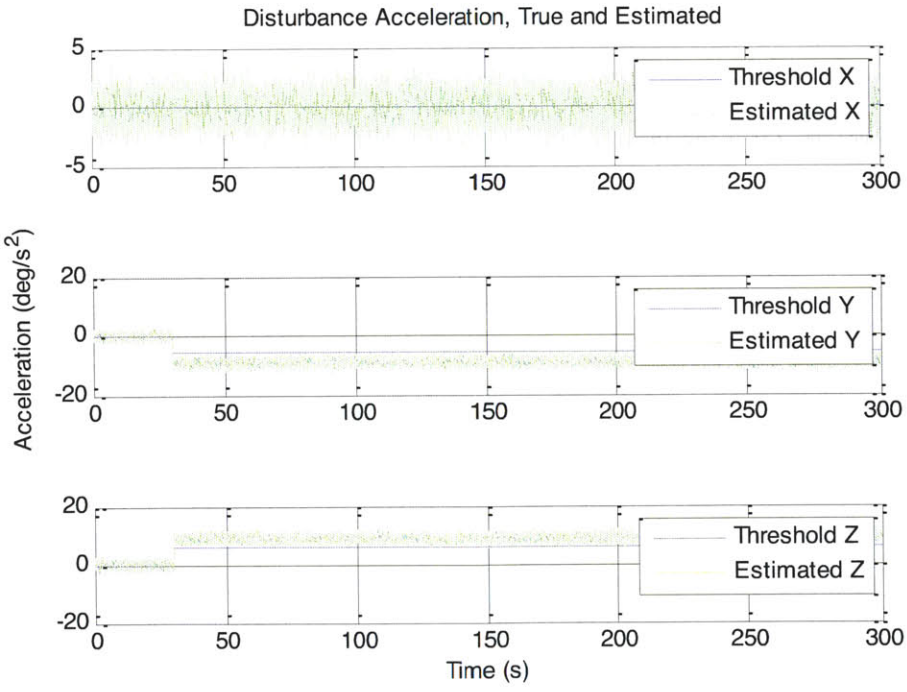


Figure 5-28: Angular Disturbance Acceleration

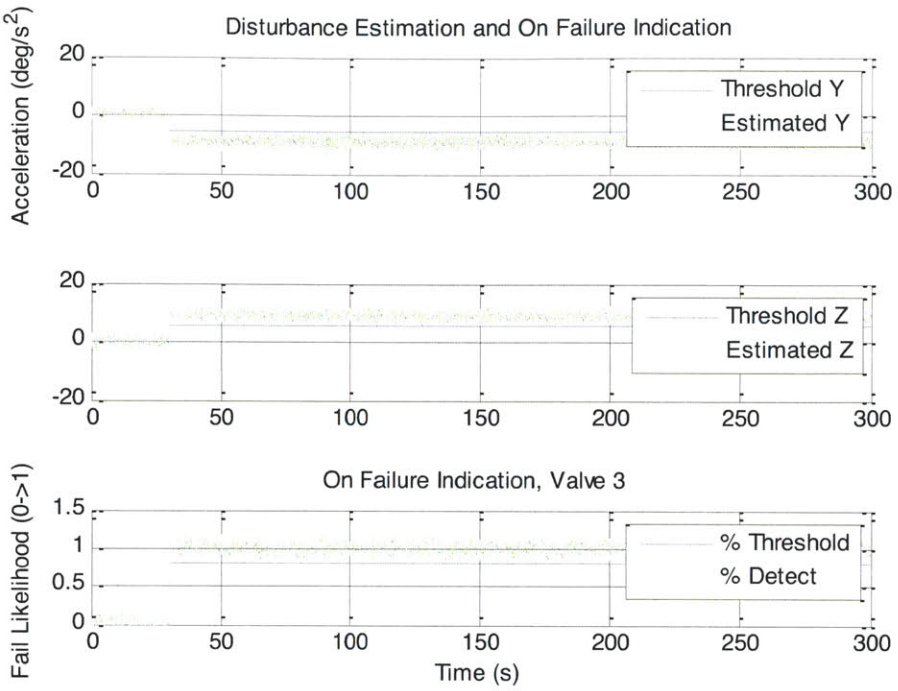


Figure 5-29: Angular Disturbance and Failure Indication

Overall the above simulation shows that the control and detection algorithms which are used are still effective at controlling the vehicle, and detecting and mitigating failures, even when the mass properties have changed significantly. The vehicle control characteristics are slightly different, as is evident from the slightly different motion of the vehicles through the maneuvers. This is most evident in the phase space regions of Figure 5-29, where the vehicle motion between regions 1, 2, and 3 is different from previous simulation cases. The simple explanation for this is that the vehicle control authority and activity vectors have changed for each jet, so therefore the vehicle will naturally move differently.

5.3.7 Asymmetric Mass Properties Not Updated in Autopilot

In the following example, the vehicle mass properties are the same as in Section 5.3.6, however the changes are not updated in the flight controls or detection algorithms. The mass property values in the autopilot and detection algorithms are the same as in Sections 5.3.1 through 5.3.5. This is representative of if no mass property estimation algorithm were included on the final vehicle, or if there were no way to measure the sample mass properties on the Mars surface prior to launch. As in Section 5.3.6, valve 3 fails on thirty seconds into the simulation.

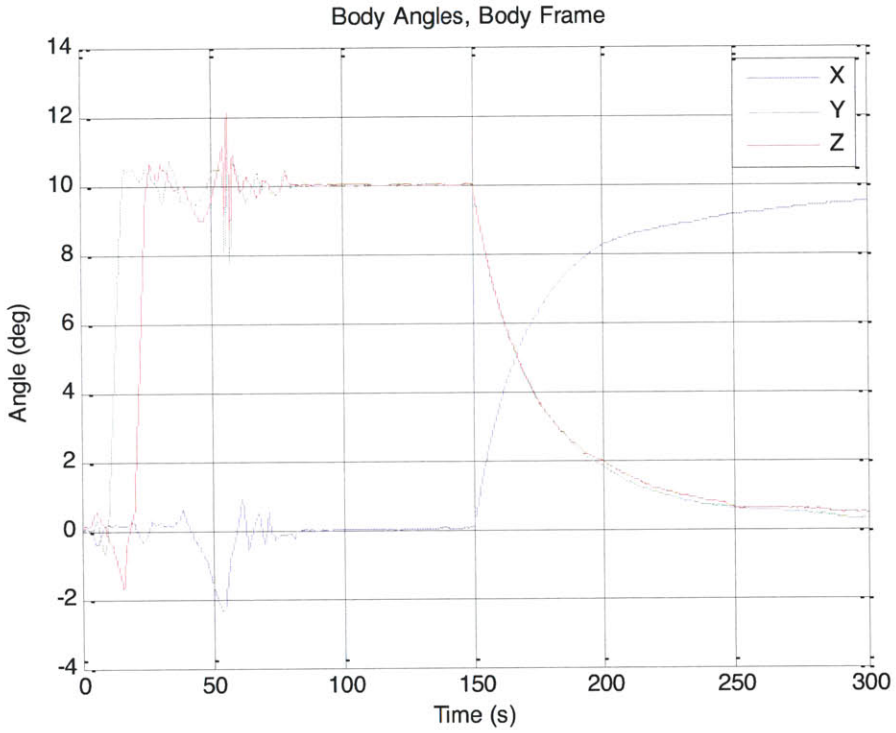


Figure 5-30: Vehicle Angular Position

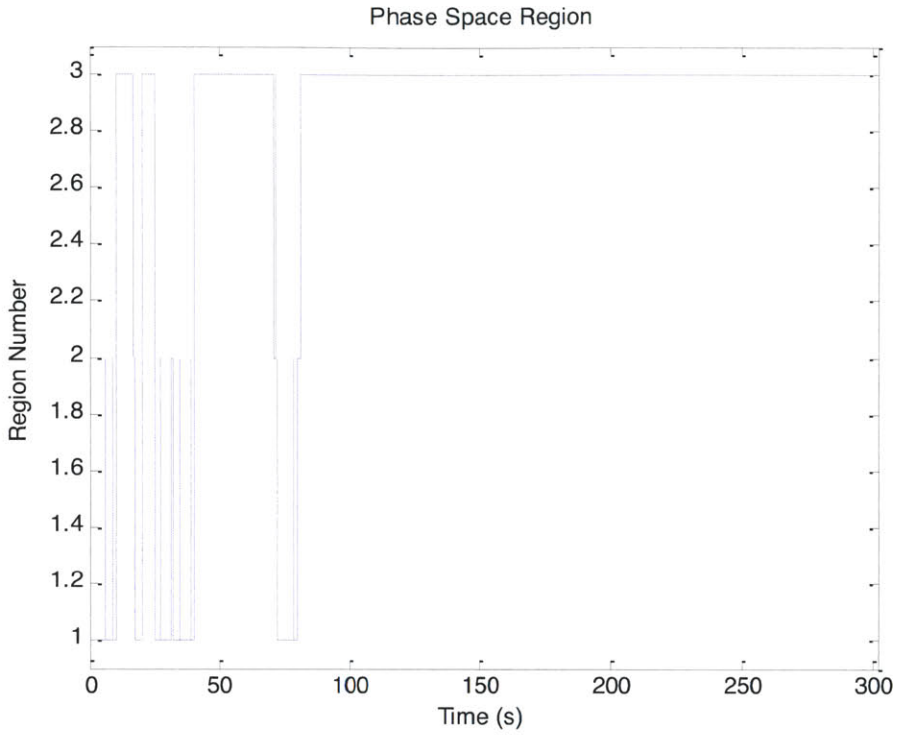


Figure 5-31: Phase Space Region

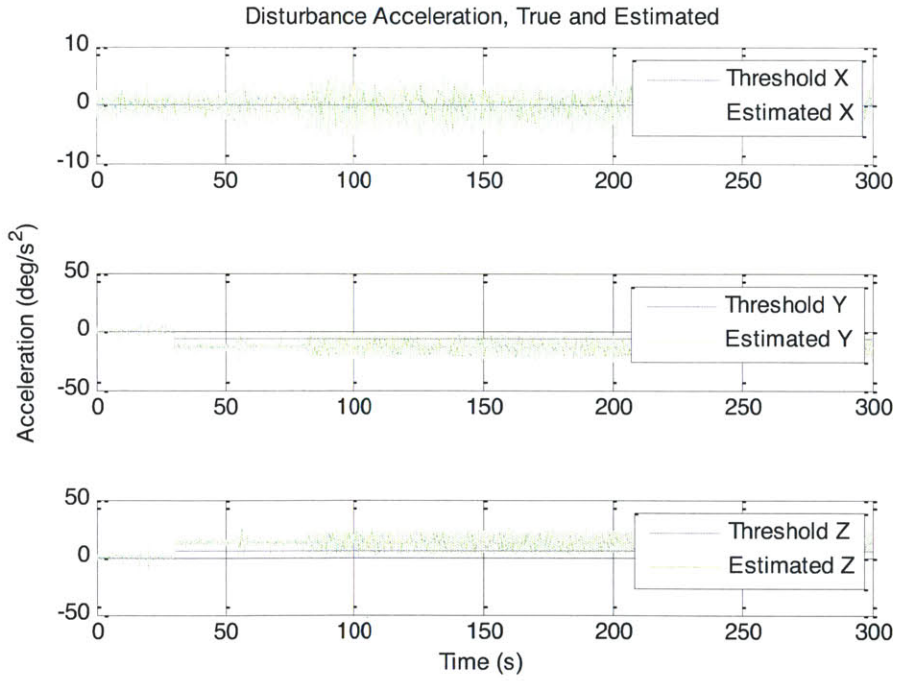


Figure 5-32: Angular Disturbance Acceleration

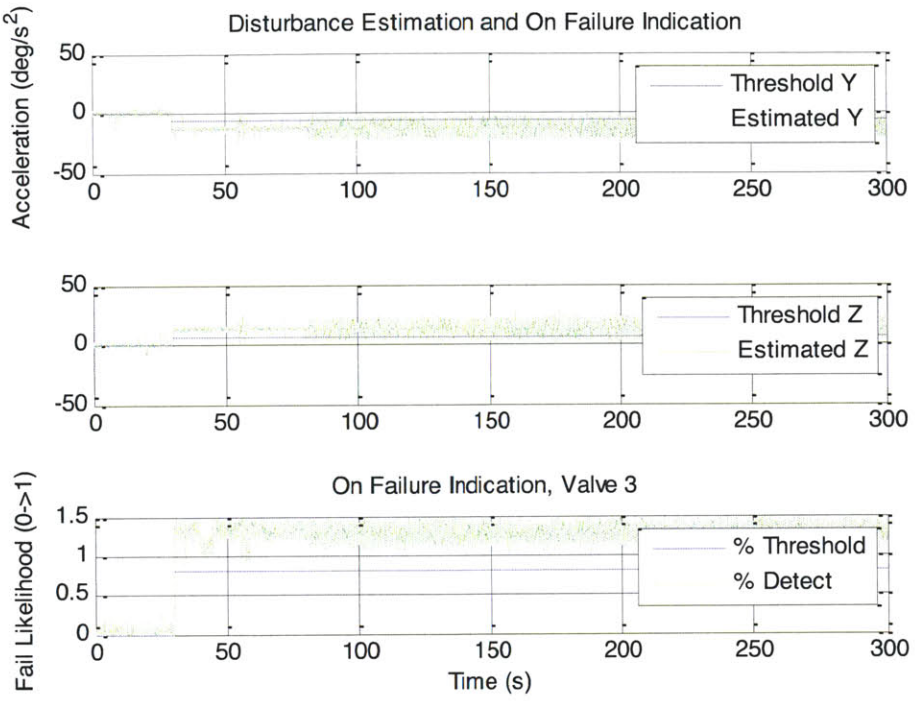


Figure 5-33: Angular Disturbance and Failure Indication

When the mass properties are not updated in the autopilot and the detection algorithms, they appear as a disturbance on the vehicle. As discussed previously, the disturbance is the difference between the vehicle's expected motion and its actual motion. The vehicle is not moving as expected, and the disturbance is amplified because the mass properties have not been updated. In this case the vehicle is still controllable and the detection algorithm is still successful, however it is not difficult to imagine a scenario in which a failure to update mass properties could prevent the vehicle from accomplishing its mission.

5.3.8 Partial On Failure or External Disturbance

An additional simulation is included to show the behavior of the phase space controller and failure algorithms in the presence of a partial on failure, which is the same as for a small external disturbance. Here valve 7 is failed on at 15% thrust.

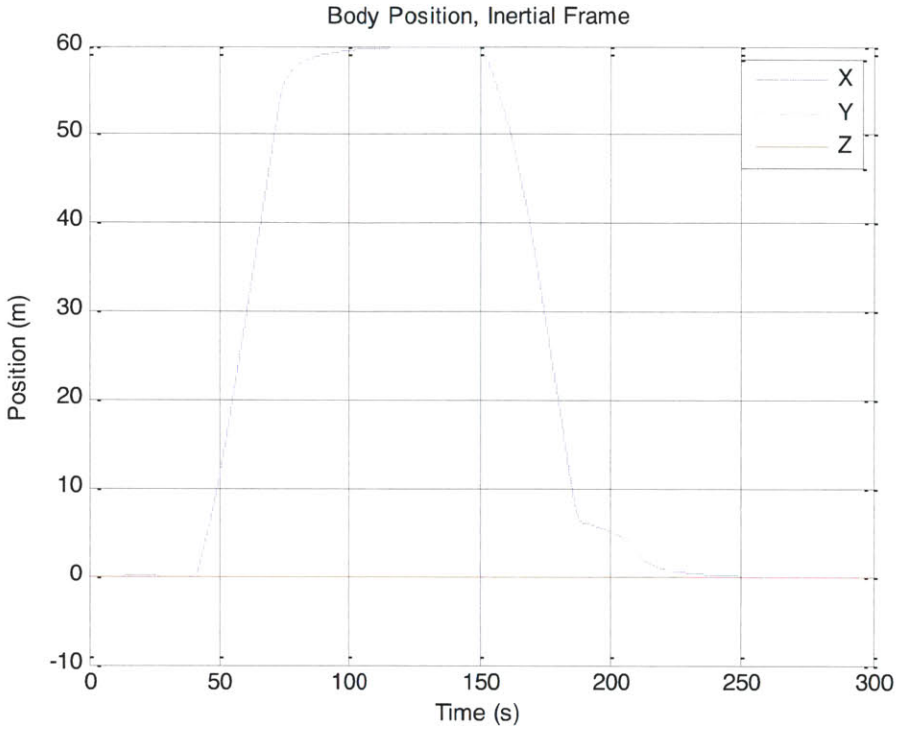
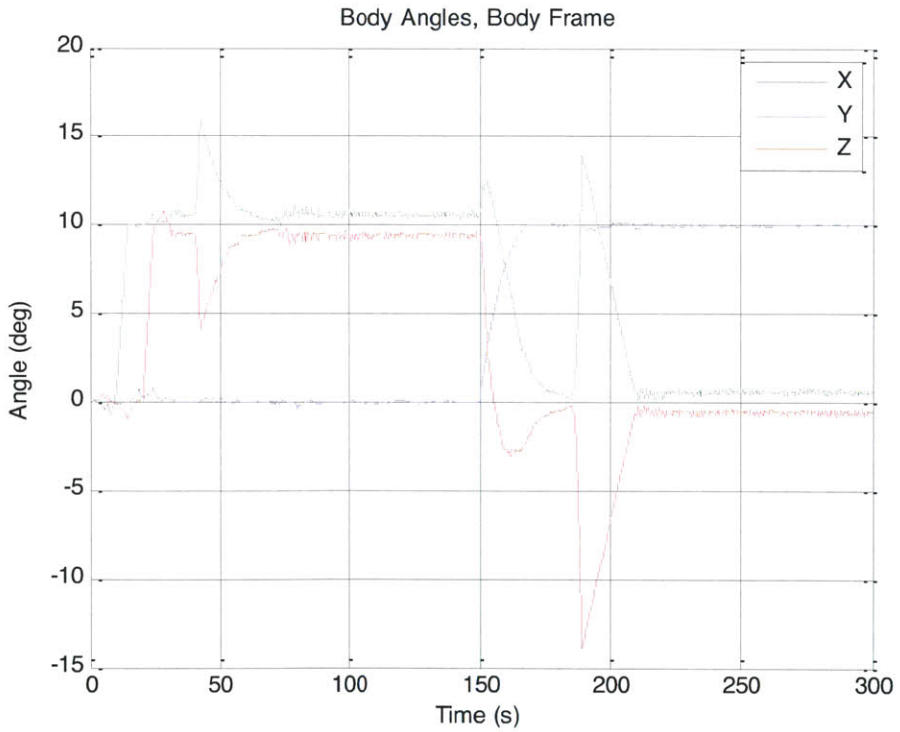
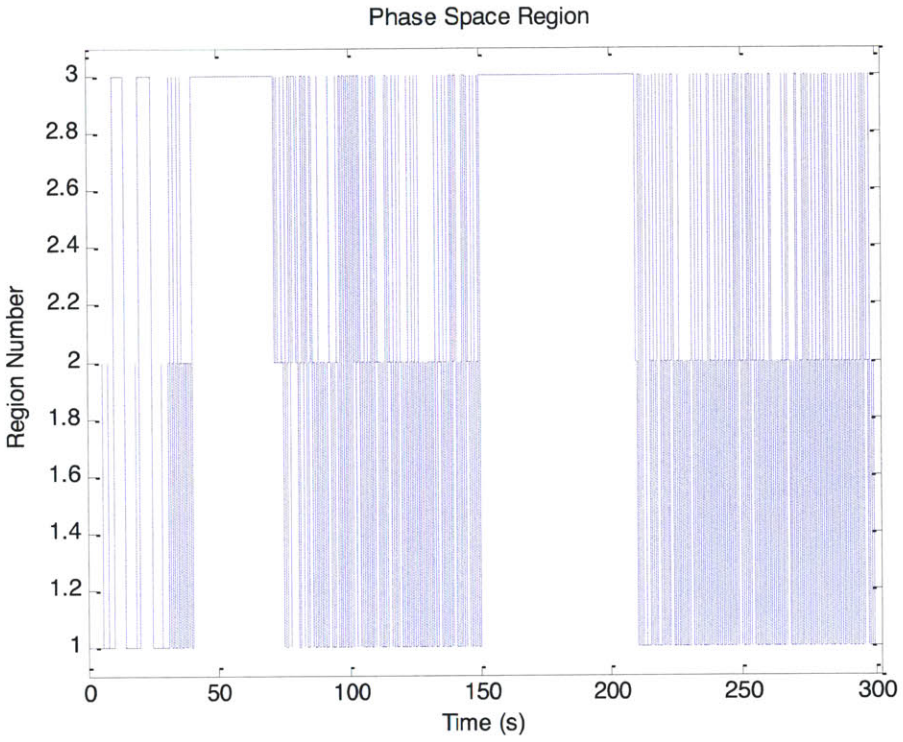


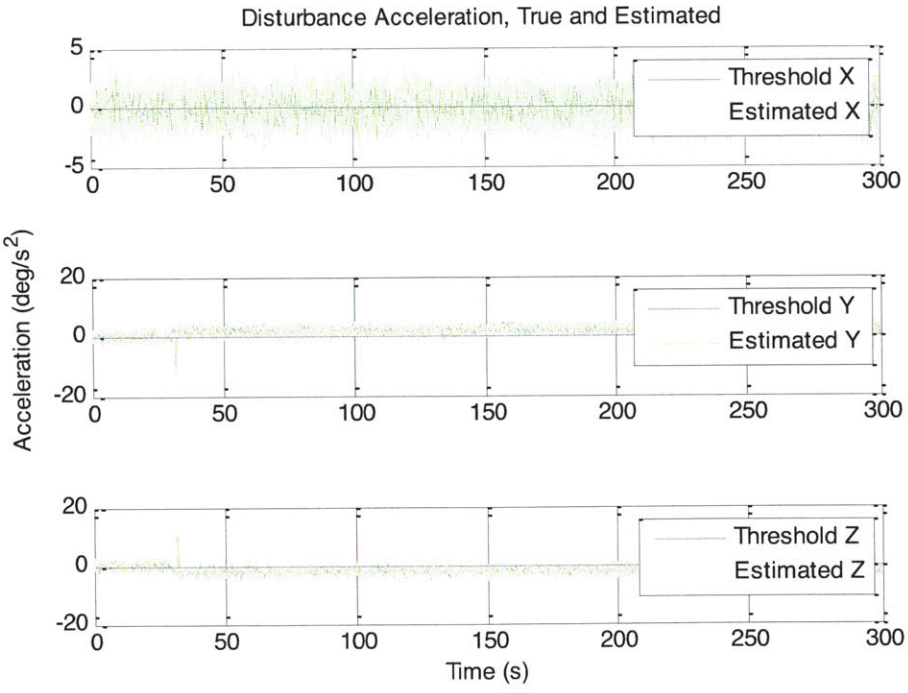
Figure 5-34: Vehicle Linear Position



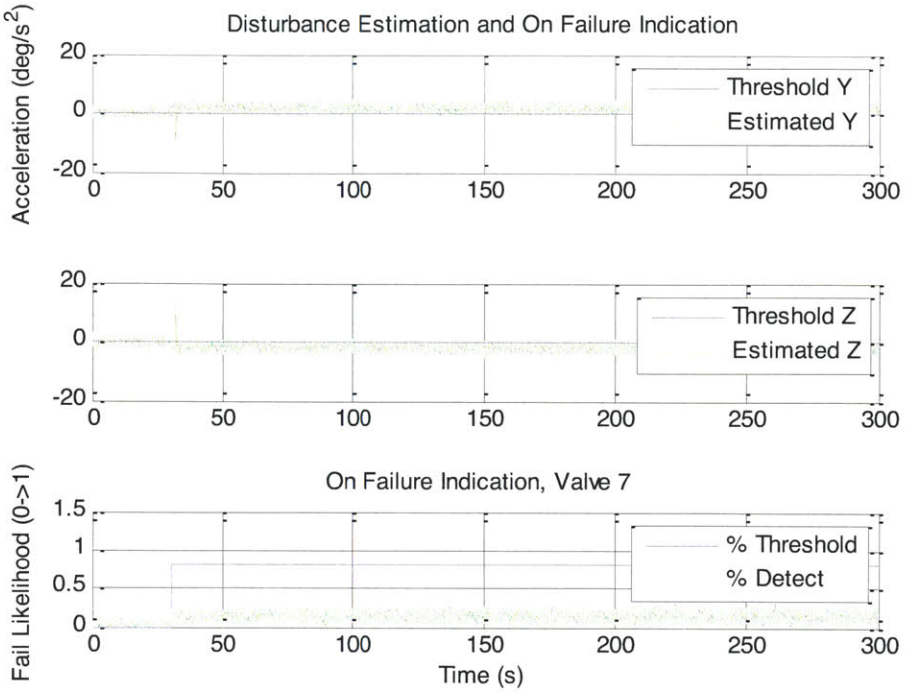
5-35: Vehicle Angular Position



5-36: Phase Space Region



5-37: Angular Disturbance Acceleration



5-38: Angular Disturbance and Failure Indication

While it is limited to disturbances significantly less than an individual jet activity vector, the phase space control law is capable of compensating for small disturbances. This performance is the same for a jet failed partially on as for a small external disturbance, such as gravity gradient or aerodynamic drag. The phase space region plot, Figure 5-36, shows the limit cycling behavior of the vehicle in response to the disturbance, where the vehicle rapidly alternates between regions 1, 2 and 3 throughout much of the simulation. It is important to note, however, that the vehicle remains controllable at all times and does successfully complete the desired maneuvers.

6.0 Conclusions and Future Work

These algorithms provide an effective means of increasing robustness and mitigating risk for a solid fuel six degree-of-freedom RCS, such as may be used on the Mars Ascent Vehicle stage of the Mars Sample Return mission. They can detect, isolate and mitigate failures without additional hardware, and with minimal computing load. This provides a significant gain in reliability and robustness to failure without adding additional cost, complexity or mass, which are all essential to any space mission. As the ambition, and therefore the size, mass, complexity, and range of unmanned space programs continues to grow over time, strategies such as this that increase mission reliability and save on expense will become increasingly relevant.

These results show that the algorithms as described can effectively detect and mitigate single valve failures in a solid fuel reaction control system. The mitigation strategy as described is very effective at reducing the effects of either an on or an off failure. An added benefit is that off-modulating the opposed jet when an on-failed jet is commanded allows for maximum possible control authority in the presence of a failure. Additionally, these algorithms are intended for computational efficiency, and are designed to be implemented on the processing-constrained, space-qualified computers which would be used for the MSR mission.

Not just applicable to the described MSR mission and MAV, this research could be applied to a wide variety of vehicles and applications in which expected and actual state changes can be differentiated to detect and isolate failures.

The most immediate future work on this topic is to find algorithms for FDI and mitigation for partial on failures and off-nominal thrust. These failure modes cannot be adequately detected and counteracted using the algorithms as described. With slight modification, however, the detection algorithm could be made to work for these cases, and the

principles of the mitigation algorithm would remain the same. Also, the described addition should be made to the algorithm to better detect intermittent off failures. A longer term future goal should be to detect multiple simultaneous failures. This should be achievable within the algorithm framework developed in this thesis, though it will require significant additional research.

An additional future topic would be to include state estimation for sensor measurement noise and measurement error. This thesis assumes that the measurements and data provided by the vehicle IMU are the best available, however every IMU will have certain noise and error characteristics, including traditional noise and other sources such as bias and gyro drift. Countermeasures for these error sources could be added to the support electronics and signal processing suite, or directly into the detection and mitigation algorithms, as necessary for the particular application. This would help to improve the overall performance of the system, especially in consideration of the limitations of modern sensors.

References

[1] D. Stephenson, "Mars Ascent Vehicle- Concept development," in *38 th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, Indianapolis, IN*, 2002.

[2] ATK Tactical Propulsion and Controls, "Orion Launch Abort System Attitude Control Motor," 15-Dec-2009. [Online]. Available: http://www.atk.com/Products/documents/ACM%20-%20Attitude%20Control%20Motor.pdf. [Accessed: 17-Jan-2012].

[3] M. Jackson and R. Gonzalez, "Orion Orbit Reaction Control Assessment," in *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007, pp. 1–16.

[4] R. F. Richfield, "Input Selection and Convergence Testing for Use with a Second Order Mass Property Estimator," Massachusetts Institute of Technology, 1985.

[5] E. Bergmann, S. Croopnick, J. Turkovich, and C. Work, "An advanced spacecraft autopilot concept," *Journal of Guidance and Control*, vol. 2, no. 3, p. 161, 1979.

[6] E. Bergmann, S. Croopnick, J. Turkovich, and C. Work, "Spacecraft flight control with the new phase space control law and optimal linear jet select," in *Guidance and Control Conference*, 1977, vol. 1, pp. 256–266.

[7] M. L. Kellogg, "Precise Nulling of Attitude and Motion Errors of a Spacecraft Using a Phase Space Autopilot," Master's Thesis, Dept. of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1978.

[8] B. S. Crawford, "Configuration Design and Efficient Operation of Redundant Multi-jet Systems," presented at the AIAA Guidance,Control and Flight Mechanics Conference, Princeton, NJ, 1969, vol. 845, p. 19.

[9] "On-Orbit Flight Control Algorithm Description," Charles Stark Draper Laboratory, Cambridge, MA, NASA-CR-151088, 1976.

[10] P. D. Hattis, "Boolean Algebrai Implementation of the Shuttle On-Orbit Autopilot Jet-Selection Algorithm," Charles Stark Draper Laboratory, Cambridge, MA, C-5188, 1979.

[11] D. R. Glandorf, "An Innovative Approach to the Solution of a Family of Linear Programming Problems," Lockheed Engineering and Management Services Company, Houston, TX, NASA Report NASA-CR-182987, Dec. 1987.

[12] E. Bergmann, "Interaction of the Space Shuttle on-orbit autopilot with tether dynamics," *Space tethers for science in the space station era;* Proceedings of the Second International Conference, Venice, Italy; 4-8 Oct. 1987, pp. 111–115, 1988.

[13] D. R. Levy, "Mass Property Estimation with Jet Failure Identification for Control of Asymmetrical Satellites," Master's Thesis, Dept. of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1984.

[14] J. J. Deyst and J. C. Deckert, "Maximum Likelihood Failure Detection Techniques Applied to the Shuttle RCS Jets," *J. Spacecr. Rockets*, vol. 13, 1976.

[15] R. M. Agustin, N. J. Adams, R. M. Hain, and R. S. Mangoubi, "Robust Failure Detection for Reentry Vehicle Attitude Control Systems," *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 6, pp. 839–845, Nov. 1999.

[16] E. Hall, "MIT's Role in Project Apollo, Volume III: Computer Subsystem," Charles Stark Draper Laboratory, Cambridge, MA, R-700, Aug. 1972.

[17] Y. Fabignon, J. Dupays, G. Avalon, F. Vuillot, N. Lupoglazoff, G. Casalis, and M. Prévost, "Instabilities and pressure oscillations in solid rocket motors," *Aerospace Science and Technology*, vol. 7, no. 3, pp. 191–200, Apr. 2003.

[18] D. D. Stephenson and H. J. Willenberg, "Mars ascent vehicle key elements of a Mars Sample Return mission," in *2006 IEEE Aerospace Conference*.